ICCBR 2015

Frankfurt, Germany 28 - 30 September 2015

Twenty-Third International Conference on Case-Based Reasoning (ICCBR 2015)

Workshop Proceedings

Joseph Kendall-Morwick (Editor)

Preface

I am pleased to present the Workshop Proceedings of the Twenty-Third International Conference on Case-Based Reasoning (ICCBR-15), held on September 28^{th} - 30^{th} in Frankfurt, Germany. This year brings a mix of both new and established workshops promising to heighten engagement and innovation in CBR, including a new installment of the long-running *CBR in Health Sciences* workshop, a second year of the successful *Case-Based Agents* workshop, and a new workshop on *Experience and Creativity*. This year also features proceedings from the eighth *Computer Cooking Contest* as well as the seventh *Doctoral Consortium* – an event in which doctoral students present and receive important feedback on their dissertation research.

I would like to thank all who contributed to the success of this workshop program, especially the authors and presenters who provided the research insights comprising the essential substance of the workshops. I would also like to thank all of the program committee members for their contributions leading to high-quality submissions, and I'd especially like to thank the workshop organizers for their hard work over the past year developing a compelling collection of workshop programs.

My special thanks go to the program chairs, Mirjam Minor and Eyke Hüllermeier, the local chairs, Eric Kübler and Jenny Quasten, and the publicity chair, Pascal Reuss, for their continuous support, efforts in planning the event, and assistance with producing the proceedings. I'd also like to thank David Leake for his support and assistance.

I hope that authors and other participants enjoy and are invigorated by this year's workshop program and also find valuable resources in these proceedings for furthering their research. I look forward to a fruitful exchange of ideas in Frankfurt.

September 2015
Frankfurt

Joseph Kendall-Morwick
(Workshop Chair)

Table of Contents

Workshop on Case-Based Agents

Workshop on Experience and Creativity

Workshop on CBR in Health Sciences

Computer Cooking Contest

Doctoral Consortium

Case-Based Agents

Workshop at the
Twenty-Third International Conference on
Case-Based Reasoning
(ICCBR 2015)

Frankfurt, Germany
September 2015

David W. Aha and Michael W. Floyd (Editors)

Chairs

David W. Aha	Naval Research Laboratory, USA
Michael W. Floyd	Knexus Research Corporation, USA

Program Committee

Klaus-Dieter Althoff	University of Hildesheim, Germany
Daniel Borrajo	University of Carlos III of Madrid, Spain
Sutanu Chakraborti	IIT Madras, India
Alexandra Coman	Ohio Northern University, USA
Amélie Cordier	Claude Bernard University of Lyon 1, France
Santiago Ontañón	Drexel University, USA
Miltos Petridis	University of Brighton, UK
Jonathan Rubin	Palo Alto Research Center, USA
Swaroop Vattam	NRC and Naval Research Laboratory, USA

Additional Reviewers

Pascal Reuss	University of Hildesheim, Germany
Viktor Ayzenshtadt	University of Hildesheim, Germany

Preface

The ICCBR 2015 Workshop on Case-Based Agents aims to highlight the challenges autonomous agents encounter and how case-based reasoning (CBR) can be used to overcome those challenges (e.g., complex environments, imperfect information, interacting with both teammates and adversaries, unexpected events). We believe a natural synergy exists between CBR and agents, and hope this workshop will highlight the recent progress in both disciplines. This serves as a sequel to the first Workshop on Case-Based Agents, which was held at ICCBR 2014 in Cork, Ireland.

The workshop program includes eight papers that explore various ways agents can leverage case-based reasoning. Two of the papers examine agents that can detect faults or discrepancies and identify their root cases. Reuss et al. explore multi-agent fault diagnosis in an aircraft domain whereas Kann et al. discuss the KRePE system, which identifies discrepancies while performing naval mine countermeasure missions. Gabel and Godehardt use CBR to predict an opponents low-level actions in simulated robotic soccer. Similarly, Frazer et al. present an error-tolerant plan matching algorithm to improve the performance of case-based plan recognition.

Fitzgerald and Goel describe their ongoing work in robotic learning by demonstration, with a specific focus on case storage and adaptation. Sánchez-Ruiz also presents an agent that observes other agents, but instead of learning to perform an observed behavior it learns to predict the outcome of battles in StarCraft. Paul and Hüellermeier describe an agent that plays the Angry Birds game. The agent learns actions to perform using random exploration and stores cases containing the best action for each encountered game state. Coman et al. propose an agent that can potentially have conflicts between its own goals and motivations, and the goals or plans supplied to it by a user. This can cause situations where the agent might rebel rather than blindly follow user commands.

Overall, we believe these papers provide a good sampling of the ways in which case-based reasoning has been used by agents and highlight recent research trends in this area. We hope that this workshop will both provide a venue for researchers in this area to meet and discuss their work, as well as provide an entry point for researchers interested in learning about case-based agents. We would like to thank everyone who contributed to the success of this workshop, including the authors, program committee, reviewers, and the ICCBR 2015 conference organizers.

September 2015 — David W. Aha
Frankfurt — Michael W. Floyd

I Know What You're Doing: A Case Study on Case-Based Opponent Modeling and Low-Level Action Prediction

Thomas Gabel and Eicke Godehardt

Faculty of Computer Science and Engineering
Frankfurt University of Applied Sciences
60318 Frankfurt am Main, Germany
{tgabel|godehardt}@fb2.fra-uas.de

Abstract. This paper focuses on an investigation of case-based opponent player modeling in the domain of simulated robotic soccer. While in previous and related work it has frequently been claimed that the prediction of low-level actions of an opponent agent in this application domain is infeasible, we show that – at least in certain settings – an online prediction of the opponent's actions can be made with high accuracy. We also stress why the ability to know the opponent's next low-level move can be of enormous utility to one's own playing strategy.

1 Introduction

Recognizing and predicting agent behavior is of crucial importance specifically in adversary domains. The case study presented in this paper is concerned with the prediction of the low-level behavior of agents in the highly dynamic, heterogeneous, and competitive domain of robotic soccer simulation (RoboCup). Case-based reasoning represents one of the potentially useful methodologies for accomplishing the analysis of the behavior of a single or a team of agents. In this sense, the basic idea of our approach is to make a case-based agent observe its opponent and, in an online fashion, i.e. during real game play, build up a case base to be used for predicting the opponent's future actions.

In Section 2, we introduce the opponent modeling problem, point to related work, and argue why knowing an opponent's next low-level actions can be beneficial. The remainder of the paper then outlines our case-based methodology (Section 3), reviews the experimental results we obtained (Section 4), and summarizes and discusses our findings (Section 5).

2 Opponent Modeling in Robotic Soccer Simulation

RoboCup [12] is an international research initiative intending to expedite artificial intelligence and intelligent robotics research by defining a set of standard problems where various technologies can and ought to be combined solving them. Annually, there are championship tournaments in several leagues – ranging from rescue tasks over real soccer-playing robots to simulated ones.

2.1 Robotic Soccer Simulation

The focus of the paper at hand is laid upon RoboCup's 2D Simulation League, where two teams of simulated soccer-playing agents compete against one another using the Soccer Server [10], a real-time soccer simulation system.

The Soccer Server allows autonomous software agents written in an arbitrary programming language to play soccer in a client/server-based style: The server simulates the playing field, communication, the environment and its dynamics, while the clients – eleven autonomous agents per team – connect to the server and are permitted to send their intended actions (e.g. a parameterized kick or dash command) once per simulation cycle to the server via UDP. Then, the server takes all agents' actions into account, computes the subsequent world state and provides all agents with (partial) information about their environment via appropriate messages over UDP.

So, decision making must be performed in real-time or, more precisely, in discrete time steps: Every 100ms the agents can execute a low-level action and the world-state will change based on the individual actions of all players. Speaking about low-level actions, we should make clear that the actions themselves are "parameterized basic actions" and the agent can execute only one of them per time step:

- $dash(x, \alpha)$ – lets the agent accelerate along its current body orientation by relative power $x \in [0, 100]$ (if it does not accelerate, then its velocity decays) into direction $\alpha \in (-180, 180]$ relative to its body orientation
- $turn(\alpha)$ – makes the agent turn its body by $\alpha \in (-180, 180]$ where, however, the Soccer Server reduces α depending on the player's current velocity in order to simulate an inertia moment
- $kick(x, \alpha)$ – has an effect only, if the ball is within the player's kick range (1.085m around the player) and yields a kick of the ball by relative power $x \in [0, 100]$ into direction $\alpha \in (-180, 180]$
- There exist a few further actions (like tackling[1], playing foul, or, for the goal keeper, catching the ball) whose exact description is beyond scope.

Given this short description of the most important low-level actions that can be employed by the agent, it is clear that these basic actions must be combined cleverly in consecutive time steps in order to create "higher-level actions" like intercepting balls, playing passes, doing dribblings, or marking players. We will call those higher-level actions *skills* in the remainder of this paper.

Robotic Soccer represents an excellent testbed for machine learning, including approaches that involve case-based reasoning. For example, several research groups have dealt with the task of learning parts of a soccer-playing agent's behavior autonomously (for instance [9, 8, 3]). In [6], as an other example, we specifically addressed the issue of using CBR for the development of a player agent skill for intercepting balls.

[1] To tackle for the ball with a low-level action $tackle(\alpha)$ means to straddle for the ball and thus changing its velocity, even if it is not in the player's immediate kick range; such an action succeeds only with limited probability which decreases the farther the ball is away from the agent.

2.2 Related Work on Opponent Modeling

Opponent modeling is an important factor that can contribute substantially to a player's capabilities in a game, since it enables the prediction of future actions of the opponent. In doing so, it also allows for adapting one's own behavior accordingly. Case-based reasoning has been frequently used as a technique for opponent modeling in multi-agent games [4], including the domain of robotic soccer [13, 1].

Using CBR, in [13] the authors make their simulated soccer agents recognize currently executed higher-lever behaviors of the currently ball leading opponent player. These include passing, dribbling, goal-kicking and clearing. These higher-level behaviors correspond to what we refer to as skills, i.e. action sequences that are executed over a dozen or more time steps. This longer time horizon allows the agent to take appropriate counter measures.

The authors of [11] also deal with the case-based recognition of skills (higher-level behaviors, to be exact the shoot-on-goal skill) executed by an opponent soccer player, focusing on the appropriate adjustment of the similarity measure employed. While we do also think opponent modeling is useful for counteracting adversary agents, we, however, disagree with these authors claiming that "in a complex domain such as RoboCup it is infeasible to predict an agent's behavior in terms of primitive actions". Instead we will show empirically that such a low-level action prediction can be achieved during an on-going play using case-based methods. To this end, the work presented in this paper is also related to the work by Floyd et al. [5] whose goal is to mimic the overall behavior of entire soccer simulation teams, be it for the purpose of analysis or for rapid prototyping when developing one's own team, without putting too much emphasis on whether the imitators yield competitive behavior.

2.3 Related Previous Work

What is the use of knowing exactly whether an opponent is going to execute a $kick(40, 30°)$ or a $dash(80, 0°)$ low-level action next? This piece of information certainly does not reveal whether this opponent's intention is to play a pass (and to which teammate) in the near future or to dribble along. Clearly, for answering questions like that the approaches listed in the previous section are potentially more useful. But knowing the opposing agent's next low-level actions is extremely useful, when knowing the *next state on the field* is essential (cf. Figure 1 for an illustration).

In [7], we considered a soccer simulation defense scenario of crucial importance: We focused on situations where one of our players had to interfere and disturb an opponent ball leading player in order to scotch the opponent team's attack at an early stage and, even better, to eventually conquer the ball initiating a counter attack. We employed a reinforcement learning (RL) methodology that enabled our agents to autonomously acquire such an aggressive duel behavior, and we successfully embedded it into our soccer simulation team's defensive strategy. So, the goal was to learn a so-called "duelling skill" (i.e. a higher-level

behavior which in the end yields a sequence of low-level actions) which made our agent conquer the ball from the ball-leading opponent.

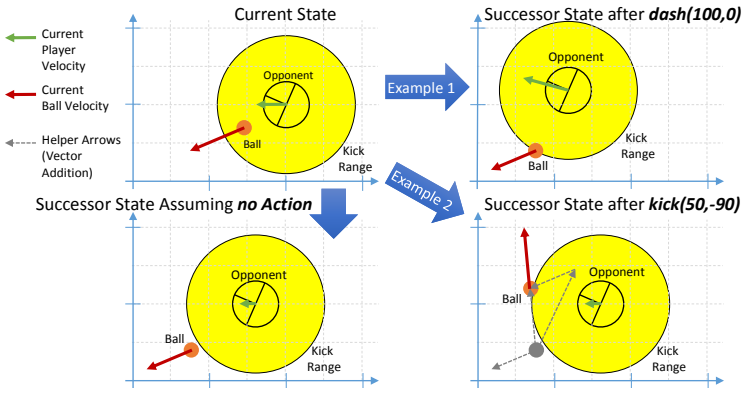


Fig. 1. In the top-left we see the current state of an opponent agent in ball possession. If we assume, this agent does not take any low-level action, then the resulting successor state looks like the one in the bottom left figure: Player and ball have moved according to their recent velocities while the magnitude of the velocity vectors have decayed according to the rules of the simulation. How different the successor state may look, if the opponent, however, does take an action (which is most likely), is shown in the right figures. In example 1 (top) the agent accelerates full power along its current body orientation, while the ball is not affected. In example 2, the player kicks the ball with 50% power into -90° relative to its current body orientation which yields a resulting ball velocity vector as shown in the bottom right.

An important feature of the soccer simulation domain is that the *model* of the environment is known. This means given, for example, the current position and velocity of the ball, it is possible for any agent to calculate the position of the ball in the next time step (because the implementation of the physical simulation by the Soccer Server is open source[2]). As a second example, when knowing one's own current position, velocity and body angle, and issuing a $turn(68°)$ low-level action, the agent can precalculate the position, velocity and body orientation it will have in the next step. Or, finally, when the agent knows the position and velocity of the ball, it can precalculate the ball's position and velocity in the next step, for any $kick(x, \alpha)$ command that it might issue.

Knowing the model of the environment (formally, the transition function $p : S \times A \times S \to \mathbb{R}$ where $p(s, a, s')$ tells the probability to end up in the next state s' when executing action a in the current state s), is extremely advantageous in reinforcement learning, since then model-based instead of model-free

[2] In practice, the Soccer Server adds some noise to all low-level actions executed, but this is of minor importance to our concerns.

learning algorithms can be applied which typically comes along with a pleasant simplification of the learning task.

So, in soccer simulation the transition function p (model of the environment) is given since the way the Soccer Server simulates a soccer match is known. In the above-mentioned "duelling task", however, the situation is aggravated: Here, we have to consider the influence of an opponent whose next actions cannot be controlled. In [7], we stated that the opponent's next (low-level) actions can "hardly be predicted [which] makes it impossible to accurately anticipate the successor state", knowing which is, as pointed out, extremely useful in RL. In the paper at hand, we will show that predicting the opponent's next low-level action might be easier than expected. As a consequence,

- in [7] we had to rely on a rough approximation of p, that merely takes into account that part of the state that can be influenced directly by the learning agent and which ignored the part of the future state which is under direct control of the ball-leading opponent (e.g. the position of the ball in the next state). This corresponded to the unrealistic assumption of an opponent that never takes any action (cf. Figure 1, bottom left).
- in future work we can employ a much more accurate version of p based on the case-based prediction of the opponent's low-level actions described in the next section.

3 Case-Based Prediction of Low-Level Actions

In what follows, we differentiate between an opponent (OPP) agent whose next low-level actions are to be predicted as well as (our) case-based agent (CBA) that essentially observes the opponent and that is going to build up a case base to be used for the prediction of OPP's actions.

When approaching the opponent modeling problem as a case-based reasoning problem, the goal of the case-based agent is to correctly predict the next action of its opponent given a characterization of the current situation. Stated differently, the current state of the system (including the case-based agent itself, its opponent as well as all other relevant objects) represents a new query q. CBA's case base $\mathcal{C}$ is made up of cases $c = (p, s)$ whose problem parts p correspond to other, older situations and corresponding solutions s which describe the action OPP has taken in situation p. Next, the case-based agent will search its case base for that case $\hat{c} = (\hat{p}, \hat{s}) \in \mathcal{C}$ (or for a set of k such cases) whose problem part features the highest similarity to the current problem q and employ its solution $\hat{s}$ as the current prediction of the opponent's next action.

3.1 Problem Modeling

In the context of this case study we focus on dribbling opponents, i.e. the opponent has the ball in its kick range and moves along while keeping the ball within its kick range all the time. Stated differently, we focus on situations

where OPP behaves according to some "dribble skill" (a higher-level dribbling behavior). Consequently, OPP executes in each time step one of the three actions $kick(x, \alpha)$, $dash(x, \alpha)$, or $turn(\alpha)$. The standard rules of the simulation allow x to be from $[0, 100]$ and α from $(-180°, 180°]$ for kicks and turns. For dashes, α is allowed to take one out of eight values (multiples of 45°). In almost all cases occurring during normal play, however, a dribbling player is heading more or less towards his opponent's goal which is why the execution of low-level turn actions represents an exceptional case. Therefore, for the time being, we leave turn actions aside and focus on the correct prediction of dashes and kicks including their parameters x and α.

Case Structure The state of the dribbling opponent (OPP) can be characterized by the x and y position of the ball within its kick range ($pos_{b,x}$ and $pos_{b,y}$) relative to the center of OPP as well as the x and y components of the ball's velocity ($vel_{b,x}$ and $vel_{b,y}$; of course, these values are also relative to OPP's body orientation). Moreover, OPP's x and y velocities ($vel_{p,x}$ and $vel_{p,y}$) are of relevance, making six features in total. The seventh relevant feature, OPP's current body orientation θ_p can be skipped due to the arguments mentioned in the preceding paragraph. Furthermore, the y component of OPP's velocity vector $vel_{p,y}$ is, in general, zero since a dribbling player almost always dribbles along its current body orientation. While this allows us to also skip the sixth feature, we remove a redundancy in the remaining features (and thus arrive at only four of them) by changing to a relative state description that incorporates some background knowledge[3] from the simulation. Hence, the problem part p of a case $c = (p, s)$ is a four-tuple $p = (pos_{bnx}, pos_{bny}, vel_{bnx}, vel_{bny})$ with

$$\begin{aligned} pos_{bnx} &= pos_{b,x} + 0.94 \cdot vel_{b,x} - 0.4 \cdot vel_{p,x} \\ pos_{bny} &= pos_{b,y} + 0.94 \cdot vel_{b,y} - 0.4 \cdot vel_{p,y} \\ vel_{pnx} &= 0.94 \cdot vel_{b,x} - 0.4 \cdot vel_{p,x} \\ vel_{pny} &= 0.94 \cdot vel_{b,y} - 0.4 \cdot vel_{p,y} \end{aligned}$$

where all components characterize the next state as it would arise, if the agent would not take any action (cf. Figure 1).

The solution s of a case $c = (p, s)$ consists of a class label l ("dash" or "kick") as well as two accompanying real-valued attributes for the power x and angle α parameters of the respective action. Thus, the solution is a triple $s = (l, x, \alpha)$.

3.2 Implementing the CBR Cycle

The case-based agent CBA observes his opponent OPP and, in doing so, builds up its case base. Note that all agents in soccer simulation act on incomplete and uncertain information. Their visual input consists of noisy information about objects in their limited field of vision. However, if the observed opponents are

[3] Knowledge about how the Soccer Server decays objects.

near and constantly focused at, CBA is provided with sufficiently accurate visual state information. In order to fill the contents of the cases' solution parts, however, CBA must apply inverse dynamics of the soccer simulation. If CBA, for example, observes that the velocity vector of the ball has been changed at time $t+1$ as in the bottom right part of Figure 1, then it can conclude that OPP has executed a $kick(50, -90°)$ action at time t and can use that information to complete the case it created at time step t.

With ongoing observation of dribbling opponent players, CBA's case base $\mathcal{C}$ grows and becomes more and more competent. Therefore, after $|\mathcal{C}|$ exceeds some threshold, CBA can utilize its case base and query it to find a prediction of the action that OPP is going to take in the current time step.

Retrieval and Similarity Measures We model the problem similarity using the local-global principle [2] with identical local similarity measures for all problem attributes, $sim_i(q_i, c_i) = (\frac{q_i - c_i}{max_i - min_i})^2$, where min_i and max_i denote the minimum and maximum value of the domain of the ith feature. The global similarity is formed as a weighted average according to

$$Sim(q, c) = \frac{\sum_{i=1}^{n} w_i \cdot sim_i(q_i, c_i)}{\sum_{i=1}^{n} w_i}$$

where attributes pos_{bnx} and pos_{bny} are weighted twice as much as vel_{pnx} and vel_{pny}.

We perform standard k-nearest neighbor retrieval using a value of $k = 3$ in our experiments. When predicting the class of the solution, i.e. the type of the low-level action (dash or kick), we apply a majority voting, and for the prediction of the action parameters (x and α) we calculate the average over all cases among the k nearest neighbors whose class label matches the majority class.

4 Experimental Results

To evaluate our approach we selected a set of contemporary soccer simulation team binaries (top teams from recent years) and made one of their agents (OPP) dribble for up to 2000 simulated time steps[4]. Our case-based agent CBA was allowed meanwhile to observe OPP and build up its case base. We evaluated CBA's performance in predicting OPP's low-level actions for increasing case base sizes.

Figure 2 visualizes the learning progress against an opponent agent from team WrightEagle. As can be seen, compelling accuracies can be achieved for both, the correctness of the type of the action (dash or kick) as well as for the belonging action parameters. Interestingly, even the relative power / angle of kicks can be predicted quite reliably with a remaining absolute error of less than ten percent / ten degrees.

[4] Duration of a regular match is 6000 time steps.

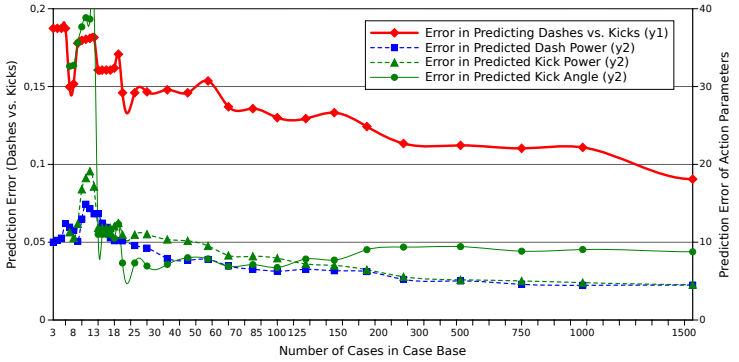


Fig. 2. Progress of CBA's competence in predicting the next low-level actions of a dribbling opponent agent from team WrightEagle. A case base of about 1500 cases was created during the course of 2000 simulated time steps.

Figure 3 focuses on different opponent agents and highlights the fact that a substantial improvement in action type prediction accuracy can be obtained with as little as 100 collected cases. Baseline to all these classification experiments is the error of the "trivial" classifier (black) that predicts each action type to be of the majority class. The right part of Figure 3 presents the recall of both, dash and kick actions. Apparently, dashes are somewhat easier to predict than kicks where, however, the recall of the latter is still above 65% for each of the opponent agents considered.

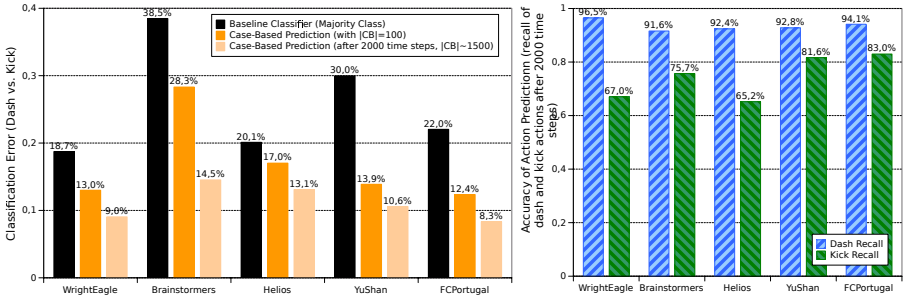


Fig. 3. Left: Case-based prediction of the type (dash or kick) of the next low-level action for opponents from different teams. Right: Recall, i.e. share of dashes that were correctly predicted as dashes and kicks that were correctly predicted as kicks.

In Figure 4, we present aggregate numbers (averages over all opponents) that emphasize how accurately the parameters of an action were predicted, given that the type of the action could be identified correctly. To this end, dash angles α are disregarded since more than 99.2% of all dash actions performed used

$\alpha = 0$, i.e. yielded a dash forward. Here, we compare (a) an "early" case base with only 10 cases, (b) an intermediate one[5] with $|\mathcal{C}| = 100$ as well as (c) one that has resulted from 2000 simulated time steps and contains circa 1500 cases. Interestingly, even in (a) comparatively low errors can be obtained. In (b) and (c), however, the resulting average absolute prediction errors become really competitive (± 2.9 for dash powers x with $x \in [0, 100]$, ± 6.3 for kick powers x with $x \in [0, 100]$, and $\pm 19.7°$ for kick angles α with $\alpha \in [0°, 360°]$).

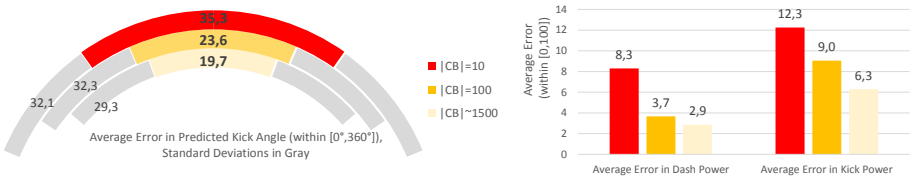


Fig. 4. Exactness of the prediction of the action parameters for different stages during ongoing learning (10, 100, and ≈1500 cases in the case base). Left: Average error of the predicted angle of a kick action. Right: Average error of the predicted relative power of a kick action and dash action (averages over agents from all opponent teams considered).

5 Discussion and Conclusion

Clearly, dribbling opponents are very likely to behave differently when they are disturbed, tackled, or attacked by a nearby opponent. Therefore, the approach presented needs to be extended to "duelling situations" as they frequently arise in real matches. For example, in scenarios like that the dribbler will presumably not just dribble straight ahead, but also frequently execute turn actions (e.g. in order to dribble around its disturber). This represents an aggravation of the action type prediction problem since then three instead of two classes of actions must be considered (dask, kick, turn).

While the case study presented focused solely on non-attacked dribbling opponents, this approach can easily be transferred to related or similar situations where knowing the opponent's next move is crucial, too. This includes, but is not limited to the behavior of an opponent striker when trying to perform a shoot onto the goal (which typically requires a couple of time steps), the behavior of the shooter as well as the goal keeper during penalty shoot-outs, or the positioning behavior of the opponent goalie (anticipating which can be essential for the striker).

As a next step, we plan to combine the presented case-based prediction of low-level actions with the reinforcement learning-based acquisition of agent behaviors as outlined in Section 2.3. This involves, first, solving the aggravated problem of

[5] A case base of a size of about 100 to 500 cases can easily be created within the first half of a match for most players.

correctly recognizing three different classes of low-level actions mentioned at the beginning of this section and, second, a proper utilization of the thereby obtained improved model when learning a higher-level duelling skill using RL. Another interesting direction for future work is the idea to let CBA start off with some opponent model in form of a case-base acquired offline (against, for example, an older version of the team to be faced) and, using appropriate techniques for case base maintenance, to successively replace old experience by new experience gained online during the current match.

References

1. Ahmadi, M., Keighobadi-Lamjiri, A., Nevisi, M., Habibi, J., Badie, K.: Using a Two-Layered Case-Based Reasoning for Prediction in Soccer Coach. In: Proceedings of the International Conference of Machine Learning; Models, Technologies and Applications (MLMTA'03). pp. 181–185. CSREA Press (2003)
2. Bergmann, R., Richter, M., Schmitt, S., Stahl, A., Vollrath, I.: Utility-Oriented Matching: A New Research Direction for Case-Based Reasoning. In: Proceedings of the 9th German Workshop on Case-Based Reasoning. pp. 264–274 (2001)
3. Carvalho, A., Cheriton, D.: Reinforcement Learning for the Soccer Dribbling Task. In: Proceedings of IEEE Conference on Computational Intelligence and Games (CIG). pp. 95–101. Seoul, South Korea (2011)
4. Denzinger, J., Hamdan, J.: Improving Modeling of Other Agents Using Stereotypes and Compactification of Observations. In: Proceedings of Third International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS). pp. 1414–1415. New York, USA (2004)
5. Floyd, M., Esfandiari, B., Lam, K.: A Case-Based Reasoning Approach to Imitating RoboCup Players. In: Proceedings of the 21st International Florida Artificial Intelligence Research Society Conference. pp. 251–256. Coconut Grove, USA (2008)
6. Gabel, T., Riedmiller, M.: CBR for State Value Function Approximation in Reinforcement Learning. In: Proceedings of the 6th International Conference on Case-Based Reasoning (ICCBR 2005). pp. 206–221. Springer, Chicago, USA (2005)
7. Gabel, T., Riedmiller, M., Trost, F.: A Case Study on Improving Defense Behavior in Soccer Simulation 2D: The NeuroHassle Approach. In: L. Iocchi, H. Matsubara, A. Weitzenfeld, C. Zhou, editors, RoboCup 2008: Robot Soccer World Cup XII, LNCS. pp. 61–72. Springer, Suzhou, China (2008)
8. Kalyanakrishnan, S., Liu, Y., Stone, P.: Half Field Offense in RoboCup Soccer: A Multiagent Reinforcement Learning Case Study. In: RoboCup-2006: Robot Soccer World Cup X. pp. 72–85. Springer Verlag, Berlin (2007)
9. Kuhlmann, G., Stone, P.: Progress in Learning 3 vs. 2 Keepaway. In: RoboCup-2003: Robot Soccer World Cup VII. pp. 694–702. Springer, Berlin (2004)
10. Noda, I., Matsubara, H., Hiraki, K., Frank, I.: Soccer Server: A Tool for Research on Multi-Agent Systems. Applied Artificial Intelligence 12(2-3), 233–250 (1998)
11. Steffens, T.: Similarity-Based Opponent Modelling Using Imperfect Domain Theories. In: Proceedings of the IEEE Symposium on Computational Intelligence and Games (CIG05)
12. Veloso, M., Balch, T., Stone, P.: RoboCup 2001: The Fifth Robotic Soccer World Championships. AI Magazine 1(23), 55–68 (2002)
13. Wendler, J., Bach, J.: Recognizing and Predicting Agent Behavior with Case-Based Reasoning. In: D. Polani and A. Bonarini and B. Browning (editors), RoboCup 2003: Robot Soccer World Cup VII. pp. 729–728. Padova, Italy (2004)

Case-based Local and Global Percept Processing for Rebel Agents

Alexandra Coman[1], Kellen Gillespie [2], Héctor Muñoz-Avila [3]

[1] Department of Electrical and Computer Engineering and Computer Science, Ohio Northern University, Ada, OH 45810
[2] Knexus Research Corporation, Springfield, VA 22314
[3] Department of Computer Science and Engineering, 19 Memorial Drive West, Lehigh University, Bethlehem, PA 18015
a-coman@onu.edu, kellen.gillespie@knexusresearch.com, hem4@lehigh.edu

Abstract. Rebel Agents are goal-reasoning agents capable of "refusing" a user-given goal, plan, or subplan that conflicts with the agent's own internal motivation. Rebel Agents are intended to enhance character believability, a key aspect of creating engaging narratives in any medium, among other possible uses. We propose to implement and expand upon a Rebel Agent prototype in eBotworks, a cognitive agent framework and simulation platform. To do so, we will make use of (1) a case-based reasoning approach to motivation-discrepancy-perception, and (2) user input for creating the agents' "emotional baggage" potentially sparking "rebellion".

Keywords: rebel agents, character believability, local and global perceptual processing

1 Introduction

Rebel Agents [6] are motivated, goal-reasoning agents capable of "refusing" a goal, plan, or subplan assigned by a human user or by another agent. This rejection is the result of a conflict arising between the given goal or plan and the agent's own internal motivation. In our previous work, we made the assumption that this motivation is modeled for the purpose of creating character believability [1], a key aspect of engaging narratives in any medium. However, different motivation models are also applicable. In the context of rebel agents, the term "motivation discrepancies" refers to incongruities between a character's motivation and the character's assigned goal and/or course of action. When a motivation discrepancy occurs, depending on the perceived intensity of the incongruity, the Rebel Agent may generate a new goal that safeguards its motivations. While so far explored in the context of interactive storytelling and character believability, the potential applications of rebel agents are by no means limited to this. Such agents can also be useful, for example, in mixed-initiative situations in which the Rebel Agent may have access to information unavailable to its human collaborator, and use this information to decide when to reject a command received from the collaborator.

We are in the process of developing a conceptual framework for Rebel Agents and implementing a Rebel Agent prototype in eBotworks, a cognitive agent framework and simulation platform [15].

In previous work [7], we explained that, for the purpose of detecting and reacting to "motivation discrepancies", eBotworks agents should be made able to perceive and interpret their surroundings in "subjective" ways potentially eliciting "emotion" intense enough to cause rebellion. We showed how eBotworks agent perception, which is by default omniscient and objective, needs to be modified to more closely mimic (or appear to mimic) human perception. We also described that this can be achieved using sensory filters informed by mechanisms of human perception. These mechanisms include gradual perception differentiation, local and global percept processing and, perhaps most importantly for our purposes, the bidirectional connection between perception and emotion. That is, perception can elicit emotion and is, in turn, affected by emotion.

While relying on psychology literature to build these filters, we are ultimately aiming for agents with believable observable behavior, but not based on complex models of cognition.

We aim to endow our prototype Rebel Agent with motivation based on emotionally-charged autobiographical memories. For example, a bot that reaches a location at which something "traumatic" happened in the past might undergo a goal change appropriate to the context. The retrieval of autobiographical memories is to initially occur based on location ecphory [14], that is, location-specific memory cues. They use exact physical locations (i.e. map coordinates) as memory cues. This choice is preferable from a practical standpoint, but does not accurately reflect the way location ecphory works in humans. The characteristics of a location that awaken memories and incite emotion tend to be the sights, sounds, smells, tastes, and tactile sensations pertaining to it, not necessarily its map coordinates. However, while location coordinates are easy to retrieve and to compare, the same cannot be said about complex combinations of percepts.

In previous work [7], we explained how the perception mechanisms of eBotworks can be modified in order to acquire percepts in a more "human-like" manner.

Herein, we approach the challenge of retrieving past percepts and comparing them to current ones using the case-based reasoning model, which is a natural match for this retrieval process. Case-based reasoning literature offers examples of complex case structures and associated similarity measures (e.g., [4][13][18]), allowing us to store and compare complex scene representations, thus taking location ecphory beyond mere map coordinates.

In building a case base consisting of "memories" of percepts and associated emotions, one of the challenges is providing the basis upon which the agents associate emotions to percepts. While this could be accomplished by building a complex inner model of the agent, herein, we discuss a knowledge-engineering-light alternative. This new approach could leverage the chat-based interface of eBotworks, through which users can give agents commands. In our context, human users would be directing the agent how to "feel" in certain situations. That way, the agent, instead of being provided with a complex program dictating how it should behave in various contexts, picks up an "emotional baggage" derived from that human user's personality (or just a "role" that the human user chooses to play). By getting input from different human users, we can produce a range of bots roughly exemplifying various personalities.

Our two contributions herein are:

(1) Exploring a case-based reasoning context for motivation-discrepancy-perception in eBotworks Rebel Agents.
(2) Proposing the use of chat-based user input for creating the agents' "emotional baggage", potentially sparking "rebellion".

Finally, it must be mentioned that, although we approach them in this specific context, local and global percept processing are applicable not only to Rebel Agents, but to any intelligent agents endowed with perception capabilities.

2 Local and Global Percept-Processing and Emotion

Gradual perception differentiation and local and global percept processing have been shown, in psychology literature, to characterize human perception. Human perception has also been shown to stand in bidirectional connection with emotion; percepts of various types can elicit emotional responses [5], while perception can be influenced by emotion and motivation as explained below [9][12][22].

Perception differentiation deals with the steps of the gradual formation of a percept.

Global-first percept processing begins with global features, with local ones becoming increasingly clear in later stages. It has been argued to be induced by positive emotions, such as happiness. Citing [17] and [20], Navon [16] sees perceptual differentiation as always "*proceeding from global structuring towards more and more fine-grained analysis*". As to what makes a feature global, rather than local, Navon describes a visual scene as a hierarchical network, each node of which corresponds to a subscene. Global scenes are higher up in the hierarchy than local ones, and can be decomposed into local ones. More recently, it seems to be agreed upon that, while a widespread tendency towards global-first processing is observed, it cannot be established as a general rule applying to all individuals at all times [22].

Local-first percept processing begins from or focuses on local features. It has been argued to be more likely when under the influence of negative emotions, such as stress and sadness. However, strong motivation has also been shown to be capable of inducing local-first processing [11]. Individuals with certain personality disorders have been hypothesized to be inclined towards local precedence. Yovel, Revelle, and Mineka [21] state that obsessive-compulsive personality disorder has been connected to *"excessive visual attention to small details"*, as well as "*local interference*": an excessive focus on small details interfering with the processing of global information. The same preference for local processing has been associated with autism spectrum disorders [10].

The tendency towards global or local processing has also been theorized to be culture-specific: certain cultures have been shown to favor local precedence [8].

Connections between perception, emotion, and motivation are discussed at length by Zadra and Clore [22]. Their survey covers the effects of emotion and mood on global vs. local perception, attention, and spatial perception.

3 Local and Global Percept Processing for Rebel Agents in eBotWorks

eBotworks [15] is a software platform for designing and evaluating communicative autonomous systems in simulated environments. "Communicative" autonomous systems are those that can interact with the environment, humans, and other agents in robust and meaningful ways, including the use of natural language. eBotworks tasks have so far been limited to path-finding and obstacle-avoidance-type tasks (Figure 1), and have not been concerned with character believability.

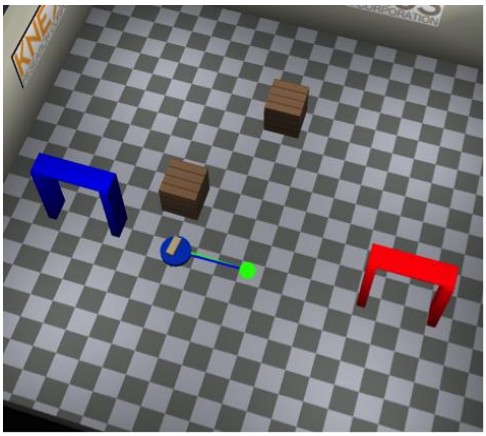


Figure 1. An eBotworks scene with an eBot performing obstacle avoidance.

In previous work [7], we designed scenarios showcasing emotion-influenced perception for possible future implementation in eBotworks. We then discussed how the implementation of these scenarios might be achieved with existing components of the framework.

We will first reiterate these scenarios before we explain how the newly-proposed mechanisms can be used to achieve them. The scenarios are based on the following assumptions: (1) the agent is a Rebel Agent [6] endowed with an autobiographical memory model in which memories are connected to emotions, (2) default perception is global-first, (3) agents have current "moods" (emotional states) which can be neutral, positive or negative, with the "neutral" mood being the default one, (4) moods can change as a result of perceiving scenes evoking autobiographical memories with emotional associations, (5) mood affects perception in the ways described in Section 2, (6) all scenarios take place on the same map, (7) in all scenarios, the agent has been assigned a goal that involves movement to a target location on the map; based on its reaction to scenes perceived on its way to the target, the agent may or may not rebel; when a rebellion threshold is reached, the agent does rebel, (8) in all scenarios, the agent perceives two scenes on its way to the target; the perception of the first scene

may or may not affect the agent's current mood, which, in turn, may influence how the second scene is perceived.

- *Scenario 1*: On the way to its target location, the agent perceives a box. This evokes no emotions, as there are no connections to the box in the autobiographical memory of the agent. Then, the agent perceives the second scene: a traffic-cone-lined driving course, using global-precedence perception. The agent's emotion changes to a slightly-positive one, as it "enjoys" driving through traffic-cone-lined driving courses. This does not elicit a goal change.
- *Scenario 2*: On the way to its target location, the agent perceives a box. In the agent's autobiographical memory, the box has positive emotional associations. This changes the agent's mood to a positive one. Positive moods favor global perception, so they do not change the agent's default perception type. The agent perceives the traffic-cone-lined driving course using global-precedence perception. The agent's mood remains positive. This does not elicit a goal change.
- *Scenario 3*: On the way to its target location, the agent perceives a box. In the agent's autobiographical memory, the box has negative emotional associations. Therefore, the agent's current mood changes to a negative one. Soon afterwards, the agent perceives the traffic-cone-lined driving course. Due to the agent's mood, local interference occurs, and the agent largely ignores the overall scene, while focusing on the color of the cones (which is similar to that of the box), which reminds it of a sad occurrence from the past, like a collision. This changes the agent's mood to a more intensely negative one, which causes the rebellion threshold to be reached and the agent to "rebel".

4 Case-Based Reasoning for Location Ecphory

Ecphory is the remembrance, caused by a memory trigger, of a past event. In the case of location ecphory, this trigger is a location with which the memory is associated.

Gomes, Martinho, and Paiva [14] use map coordinates as location-ecphory triggers. While this is easier from a practicality standpoint, the authors admit it does not accurately reflect the way location ecphory works in humans. Location coordinates (unless physically perceived, with some emotional associations) are unlikely to awaken memories and incite strong emotion. Instead, it is the sights, sounds, smells, tastes, and tactile sensations pertaining to a place that work to achieve this recollection. Thus, if these traits change beyond recognition, the location's function as a memory cue is invalidated.

Retrieving stored memories is a natural match for the case-based reasoning model, which was inspired by the psychological mechanisms underlying memory storage and recollection. Furthermore, case-based reasoning literature contains ample coverage of similarity measures between complex case structures that are not trivially comparable, including graphs, plans, and case structures based on object orientation, which is precisely what we need for implementing our Rebel Agent prototype in eBotworks. By

using case-based reasoning similarity measures, we intend to expand location ecphory beyond just map coordinates.

4.1 Case Structure and Similarity Measures

To achieve the emotional location-ecphory effect we are aiming for, each case should contain two essential pieces of information: (1) a past percept, and (2) an emotional reaction associated with that percept.

The ways in which we model these pieces of information can vary in complexity. The percept can be a complex scene or a very specific subscene, such as an individual object or something slightly more general such as a set of objects on a table. The emotional reaction can consist of a simple, basic emotion (e.g. "joy") or of a complex, layered conglomerate of emotions, each experienced at a different degree of intensity.

Due to the characteristics of our simulation platform, we are, for now, focusing on visual ecphory triggers, although triggers of a different nature (e.g. gustatory and olfactory) certainly function in the real world.

In choosing our case structure, we are influenced by the description that Navon [16] gives of a visual scene as a hierarchical network, each node of which corresponds to a subscene. Global scenes are higher up in the hierarchy than local ones, and can be decomposed into local ones. Global-first processing proceeds from global scenes, local-first processing from local ones. We do not, however, aim at matching any psychological model of perception differentiation perfectly through our case representation.

To approximate this hierarchical structure, we propose a model inspired by object-oriented ([3][2] - Section 4.4) and graph-based ([19][2] - Section 4.5) case structures.

A scene hierarchy is not equivalent to a class inheritance hierarchy, though there are clear similarities between the two. The reason is that in a class hierarchy, classes lower down in the hierarchy incorporate the attributes of classes higher up, whereas in the scene/subscene hierarchy, the inverse takes place: the root scene incorporates information from all lower nodes, because the complete scene is composed of all subscenes.

It is to be noted that the rather simple description above does not accurately capture human perception, in which a global scene is perceived as a general outline with vague details that become clear while travelling downwards in the hierarchy. Therefore, the details in the lower nodes are then incorporated (potentially completely) into the higher nodes. If perception proceeds in a global-first manner and is not prolonged, these lower levels may not be reached.

The similarity methods of Bergmann and Stahl [3] allow objects at different levels in the class hierarchy to be compared. This is especially useful, as we have no guarantees that two subscenes we are comparing are at similar hierarchical levels.

However, our situation is even more challenging: not only are the scenes that we are comparing different and at different hierarchical levels, but even their respective hierarchies can be different and correspond to varied scenes (unless the scenes that can be perceived are highly controlled and limited). Despite this challenge, we believe that the local and global similarity measures proposed by Bergmann and Stahl [3] can be adapted to be used for local and global perception, respectively. The perception setting of the agent at a given time (e.g. global after perceiving the box in Scenarios 1

and 2; local after perceiving the box in Scenario 3) will determine where in the hierarchy we look for the similarity.

For simplification, we can assume that the cases are collections of objects, and do not take into account the spatial positioning of the objects in a scene in relation to one another.

4.2 Populating the Case Base

In order to populate a case base consisting of "memories" of percepts and associated emotions, we must first provide a mechanism allowing agents' percepts to be associated with emotions.

Truly human-like agents would be able to generate emotions themselves. This would be partially based on (1) the personality with which the agent would have been endowed (which could dictate, for example, that the agent is not easily frightened), and (2) general rules about ways in which people tend to react to certain situations (e.g. a gruesome scene tends to cause shock). Thus, making agents able to generate emotions in response to percepts would require providing them with at least one of these two models.

We are interested in exploring a knowledge-light alternative to this challenge. This approach can leverage the chat interface of eBotworks (or alternative eBotworks mechanisms) and is based on the idea of having human users direct the agent on how to "feel" in certain situations. Thus, the agent acquires an "emotional baggage" derived from that human user's personality or a "role" that the human user chooses to play. Some bots, for instance, could be directed to be more "impressionable" than others.

Let us re-examine Scenario 3, where the agent perceives a box with negative emotional associations. With this approach, this association would not exist because the bot previously got hurt in the vicinity of the box, but rather because the bot was previously told that the box should make it "feel sad".

While we only propose this mechanism for the purpose of attaching specific emotions to scenes, it could later be applied more broadly within the context of motivation discrepancies and Rebel Agents. For instance, it could also be used to assign meaning to scenes, so that the agent can match scenes similar in meaning (e.g. "a quarrel") rather than just in their constitutive elements. With this ability, agents can then match emotion to meaning (e.g. witnessing a quarrel causes stress), rather than just to specific scenes and subscenes.

Currently, the chat interface of eBotworks is used to issue commands to agents in a simulated environment. For example, a user can enter "Go here" and click on a location on the current map; if the command is successful, the agent reacts by moving to the specified location.

To explain how this system could be used for our purposes, let us first assume that the bot is facing a scene containing a box. One option would be for the user to simply say one of several words corresponding to several emotions "understood" by the system, e.g. "sad". In this case, the agent would take a "snapshot" of the scene it is facing and store it together with the associated emotion, sadness.

However, memories of strong emotions can be associated with very specific subscenes, rather than to an entire complex scene (e.g. excitement associated with a logo on the envelope containing a college acceptance letter). Moreover, the subscene

that attention ends up focusing on in such situations is not necessarily related to the emotion itself. Instead, it could contain items that just happen to be there when the emotionally charged event occurs (e.g., a cup that happens to be on a nearby table while a severe argument is taking place).

To handle this possibility, we can allow the user to specify an object in the scene to which to associate the emotion by clicking on the object first, then saying the word corresponding to the emotion. In Scenario 3, clicking on a box then saying "sad" can cause the agent to switch to a sad mood and experience local interference in perception. Another necessary addition to typical eBotworks usage will be to have the agent convey, through console messages, (and, later, possibly, through visual representations on the map) what objects it is currently focusing on and what moods it is experiencing. This will enhance believability by providing insight into the agent's motivations and into the emotional justification behind its actions.

5 Conclusions

We have discussed using the case-based model for the purpose of creating location-ecphory-based motivation-discrepancy mechanisms for Rebel Agents, addressing the challenge of retrieving emotionally-charged past percepts and comparing them to current ones.

Our two main contributions herein are:

(1) Exploring a case-based reasoning context for motivation-discrepancy-perception in eBotworks Rebel Agents.
(2) Proposing the use of chat-based user input for creating the agents' "emotional baggage", potentially sparking "rebellion".

Acknowledgements. We thank the reviewers for their comments and suggestions, some of which have been integrated into the final version of the paper, others of which will be of help to us in our future work.

References

1. Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM*, 37(7), 122-125.
2. Bergmann, R. (2002). Experience management: foundations, development methodology, and Internet-based applications. Springer-Verlag.
3. Bergmann, R., & Stahl, A. (1998). Similarity measures for object-oriented case representations (pp. 25-36). Springer Berlin Heidelberg.
4. Börner, K. (1994). Structural similarity as guidance in case-based design. In Topics in case-based reasoning (pp. 197-208). Springer Berlin Heidelberg.
5. Clore G.L., & Ortony A. (2008) Appraisal theories: How cognition shapes affect into emotion. In *Handbook of Emotion 3,* 628–642 New York: Guilford Press..

6. Coman, A., & Muñoz-Avila, H. (2014). Motivation discrepancies for rebel agents: Towards a framework for case-based goal-driven autonomy for character believability. *Proceedings of the 22nd International Conference on Case-Based Reasoning (ICCBR) Workshop on Case-based Agents*.
7. Coman, A, Gillespie, K., and Muñoz-Avila, H. (2015). Believable Emotion-Influenced Perception: The Path to Motivated Rebel Agents, *Proceedings of the Advances in Cognitive Systems (ACS) Workshop on Goal Reasoning*, Atlanta, Georgia, May 28th - 31st.
8. Davidoff, J., Fonteneau, E., & Fagot, J. (2008) Local and global processing: Observations from a remote culture. *Cognition*;108(3):702–709.
9. Easterbrook J. (1959) The effect of emotion on cue utilization and the organization of behavior. *Psychol. Rev.*, 66(3):183–201.
10. Frith, U. (1989) *Autism: Explaining the enigma*. Oxford, UK: Blackwell Scientific Publications.
11. Gable P.A., & Harmon-Jones E. (2008) Approach-motivated positive affect reduces breadth of attention. *Psychol. Sci*., 19:476–482.
12. Gasper K, & Clore G.L. (2002) Mood and global versus local processing of visual information. *Psychol. Sci.*, 13:34–40
13. Goel, A. K., & Craw, S. (2005). Design, innovation and case-based reasoning. The Knowledge Engineering Review, 20(03), 271-276.
14. Gomes, P. F., Martinho, C., & Paiva, A. (2011). I've Been Here Before! Location and Appraisal in Memory Retrieval. *Int. Conf. on Autonomous Agents and Multiagent Systems*.
15. Gupta K. M., & Gillespie K. (2015) eBotworks: A software platform for developing and evaluating communicative autonomous systems. AUVSI Unmanned Systems, Atlanta, GA.
16. Navon, D. (1977). Forest before trees: The precedence of global features in visual perception. *Cognitive Psychology*. 9(3):, 353–383.
17. Palmer, S.E. (1975) Visual perception and world knowledge: Notes on a model of sensory-cognitive interaction. In D. A. Norman , D. E. Rumelhart, and the LNR Research Group, *Explorations in cognition*, San Francisco, CA: Freeman.
18. Smyth, Barry, and Padraig Cunningham. (1992) Déjà Vu: A Hierarchical Case-Based Reasoning System for Software Design. ECAI. Vol. 92.
19. Vattam, S., Aha, D.W., & Floyd, M. (2014). Case-based plan recognition using action sequence graphs. In Proceedings of the Twenty-Second International Conference on Case-Based Reasoning, (pp. 495-510). Cork, Ireland: Springer.
20. Winston, P.H. (1973) Learning to identify toy block structures. In R.L. Solso (Ed.) *Contemporary issues in cognitive psychology: The Loyola Symposium*, Washington D.C.

21. Yovel I., Revelle W., & Mineka S. (2005). Who sees trees before forest? The obsessive compulsive style of visual attention. *Psychological Science*, 16, 123-129.
22. Zadra J.R., & Clore G.L. (2011) Emotion and perception: The role of affective information. *Wiley Interdisciplinary Reviews: Cognitive Science*.

Predicting the Outcome of Small Battles in StarCraft *

Antonio A. Sánchez-Ruiz

Dep. Ingeniería del Software e Inteligencia Artificial
Universidad Complutense de Madrid (Spain)
antsanch@fdi.ucm.es

Abstract. Real-Time Strategy (RTS) games are popular testbeds for AI researchers. In this paper we compare different machine learning algorithms to predict the outcome of small battles of marines in StarCraft, a popular RTS game. The predictions are made from the perspective of an external observer of the game and they are based only on the actions that the different units perform in the battlefield. Our empirical results show that case-based approaches based on k-Nearest Neighbor classification outperform other standard classification algorithms like Linear and Quadratic Discriminant Analysis or Support Vector Machines.

Keywords: Prediction, StarCraft, Linear and Quadratic Discriminant Analysis, Support Vector Machines, k-Nearest Neighbors

1 Introduction

Real-Time Strategy (RTS) games are popular testbeds for AI researchers [4] because they provide complex and controlled environments in which to carry out different experiments. In this paper we assume the role of an external observer of the game that tries to predict the outcome when the armies of two different players engage in combat. As a spectator of the game, we can only base the predictions on the actions of the different units in the battlefield. From this perspective, we can consider each army as a group of agents working in a coordinated manner to defeat the other army. We know that the units in the game are not really agents because they are not autonomous (in fact they are controlled by a human player or by the internal AI of the game), but from the perspective of an external observer we only see several units performing actions in a simulation, and we do not know whether those actions are consequence of individual decisions or some superior intelligence. Therefore, our approach to prediction in RTS games could be applied as well to multi-agent simulations.

The ability to predict the outcome of battles is interesting because it can be used to dynamically modify the strategy of the player. For example, the player could decide to change the composition of the army, to bring more troops into

* Supported by Spanish Ministry of Economy and Competitiveness under grant TIN2014-55006-R

Fig. 1: Screenshot of our custom map: Battle of Marines

the battlefield or deploy them differently, or even to flee if the prospects are not good. In general, an agent able to make predictions (running an internal simulation based on what he knows) might be able to adapt his behavior more successfully than other agent without this ability.

In this work, we compare classical classification algorithms like Linear and Quadratic Discriminant Analysis, Support Vector Machines, and two instance-based classifiers based on the retrieval of the k-Nearest Neighbors (kNN). kNN classifiers can be seen as simple Case-based Reasoning (CBR) systems that only implement the retrieval phase of the CBR cycle. In this paper we study the accuracy of the prediction during the course of the battle, the number of games that each algorithm needs to learn, and the stability of the prediction over time.

The rest of the paper is organized as follows. Sections 2 and 3 describe the scenario used in the experiments, the process to extract the data for the analysis and the features chosen to represent the game state. Sections 4 and 5 explain the different classification algorithms and the results obtained. The paper concludes with the related work, and some conclusions and directions for future research.

2 Battles of Marines in StarCraft

StarCraft[1] is a popular RTS game in which players have to harvest resources, develop technology and build armies combining different types of units to defeat

[1] http://us.blizzard.com/en-us/games/sc/

game	frame	units1	life1	area1	units2	life2	area2	distance	winner
1	0	6	40	2772	6	40	2520	1309.903	1
1	3	6	40	2772	6	40	2520	1309.903	1
1	6	6	40	2736	6	40	2925	1302.857	1
1	9	6	40	2964	6	40	2923	1282.317	1
1	12	6	40	3876	6	40	2905	1266.277	1
1	15	6	40	4332	6	40	3045	1246.241	1

Table 1: Examples of game states extracted from a Starcraft game trace.

the other players. The combination of different types of units and abilities, and the dynamic nature of the game force players to develop strategies at different levels. At the *macro* level, players have to decide the amount of resources invested in map exploration, harvesting, technology development, troops, offensive and defensive forces, among others. At the *micro* level players have to combine different types of units, locate them in the map and use their abilities. In this paper we focus on small battles, that is, at the micro level.

StarCraft also provides a map editor to create custom games. Using this tool, we have created a simple combat scenario (Figure 1) in which each player controls a small army of 6 *terran marines* (marines are basic ground combat units with ranged attack). The game always begins with the same initial configuration, each army located on opposite sides of the map, and the game ends when all the units of one player are destroyed. In this type of scenario it is very important to strategically locate the units on the map to take advantage of the range attack and concentrate the fire on a few units to destroy them as soon as possible.

3 Data Collection and Feature Selection

In order to obtain the data to train the different classifiers, we played 200 games collecting traces that describe the evolution of the games. We configured the map so the internal game AI controls both players so (1) we can automatically play as many games as required, and (2) we know that all the games are well balanced (since the StarCraft AI is playing against itself). Finally, there is a third player that only observes the game (it does not intervene) and extracts the game traces to a file so they can be analyzed later[2].

The data set contains *traces* of 200 games in which player 1 won 119 times and player 2 the remaining 81. They are very fast games with an average duration of 19.12 seconds. In each trace we store the *game state* 6 times per second, so each game is described with approximately 114 games states or *samples*.

Each game state is stored using a vector of features (Table 1) that represents the combat power of each army and the strategic deployment of the troops in the map. The combat power is represented using the number of units alive in each

[2] We use the BWAPI framework to extract information during the game (https://github.com/bwapi/bwapi).

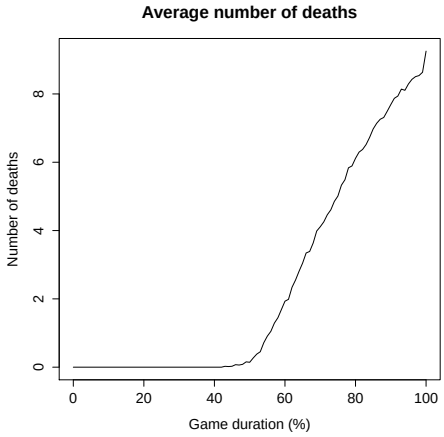


Fig. 2: Average number of deaths during the game.

army and their average life. To represent the strategic distribution of troops in the map we compute the area of the minimum axis aligned rectangle containing the units of each player and the distance between the centers of both rectangles. The rectangle area is a measure of the dispersion of the units, and the distance between the centers indicates how close the two armies are. Each game state is labeled later with the winner of that particular game. The table also shows the game and the current frame for clarity (1 second is 18 game frames although we only sample 6 of them), but we do not use those values in the prediction.

The features to describe the strategic distribution of the troops in the map are especially important during the first seconds of the game. Figure 2 shows the average number of dead units during the 200 games. As we can see, during the first half of the game the armies are approaching each other and the fight does not start until the second half. Thus, during the first seconds of the game the predictions will depend only on the relative location of the units.

4 Classification algorithms

We will compare the following classification algorithms in the experiments:

- Linear Discriminant Analysis (LDA) [8] is classical classification algorithm that uses a linear combination of features to separate the classes. It assumes that the observations within each class are drawn from a Gaussian distribution with a class specific mean vector and a covariance matrix common to all the classes.
- Quadratic Discriminant Analysis (QDA) [9] is quite similar to LDA but it does not assume that the covariance matrix of each of the classes is identical, resulting in a more flexible classifier.

Classifier	Accuracy	Parameters
Base	0.5839	
LDA	0.7297	
QDA	0.7334	
SVM	0.7627	kernel = radial, C = 1, sigma = 0.3089201
KNN	0.8430	k = 5
KKNN	0.9570	kernel = optimal, kmax = 9, distance = 2

Table 2: Classification algorithms, configuration parameters and global accuracy.

- Support Vector Machines (SVM) [7] have grown in popularity since they were developed in the 1990s and they are often considered one of the best *out-of-the-box* classifiers. SVM can efficiently perform non-linear classification using different kernels that implicitly map their inputs into high-dimensional feature spaces. In our experiments we tested 3 different kernels (lineal, polynomial and radial basis) obtaining the best results with the radial basis.
- k-Nearest Neighbour (kNN) [2] is a type of instance-based learning, or lazy learning, where the function to learn is only approximated locally and all computation is deferred until classification. The kNN algorithm is among the simplest of all machine learning algorithms and yet it has shown good results in several different problems. The classification of a sample is performed by looking for the k nearest (in Euclidean distance) training samples and deciding by majority vote.
- Weighted K-Nearest Neighbor (kkNN) [10] is a generalization of kNN that retrieves the nearest training samples according to Minkowski distance and then classifies the new sample based on the maximum of summed kernel densities. Different kernels can be used to weight the neighbors according to their distances (for example, the *rectangular* kernel corresponds to standard un-weighted kNN). We obtained the best results using the *optimal* kernel [14] that uses the asymptotically optimal non-negative weights under some assumptions about the underlying distributions of each class.

The three first algorithms use the training data (labeled game states in our experiments) to infer a generalized model, and then they use that model to classify the test data (new unseen game states). The last two algorithms, however, use the training data as cases and the classification is made based on the most similar stored cases. All the experiments have been run using the R statistical software system [11] and the algorithms implemented in the packages *caret*, *MASS*, *e1071*, *class* and *kknn*.

5 Analyzing the results

Table 2 shows the configuration parameters used in each classifier and its accuracy computed as the ratio of samples (game states) correctly classified. The optimal parameters for each classifier were selected using repeated 10-fold cross

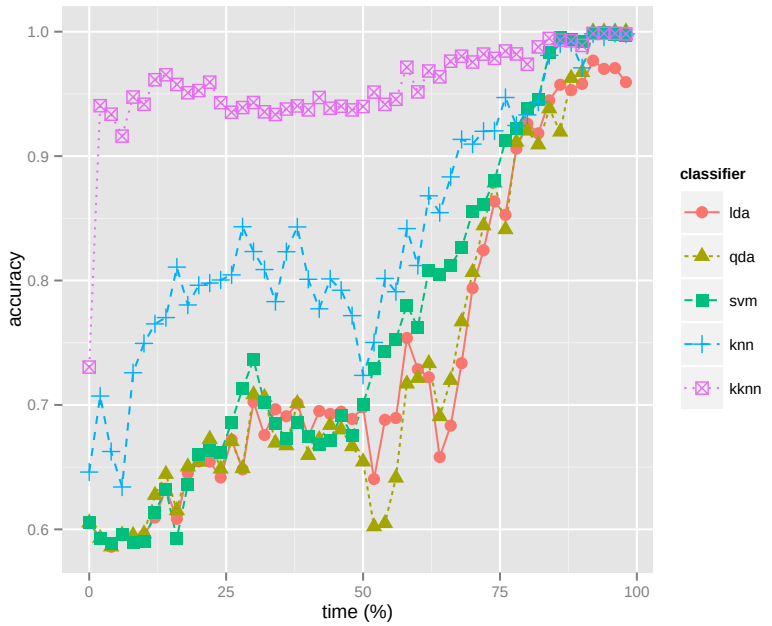


Fig. 3: Accuracy of the classifiers during the game.

validation over a wide set of different configurations. The accuracy value has been calculated as the average of 32 executions using 80% of the samples as the training set and the remaining 20% as the test set.

The *base* classifier predicts the winner based only on the proportion of samples belonging to each class (58.39% of the samples correspond to games won by player 1) and it is useful only as a baseline to compare the other classifiers. LDA, QDA and SVM obtain accuracy values ranging from 72% to 76%. The two instance-based algorithms, on the other hand, obtain higher precision values. It is especially impressive the result of kkNN that is able to predict the winner 95.70% of the times. These results seem to indicate that, in this particular scenario, it is quite difficult to obtain a generalized model, and local based methods perform much better.

The global accuracy value may not be informative enough because it does not discriminate the time of the game when the prediction is made. It is reasonable to expect the accuracy of the predictions to increase as the game evolves as it is shown in Figure 3. The x-axis represents the percentage of elapsed time (so we can uniformly represent games with different duration) and the y-axis the average accuracy of each classifier for game states from that time interval.

Selecting a winner during the second half of the game is relatively easy since we can see how the battle is progressing, but during the first half of the game the prediction problem is much more difficult and interesting since we only see

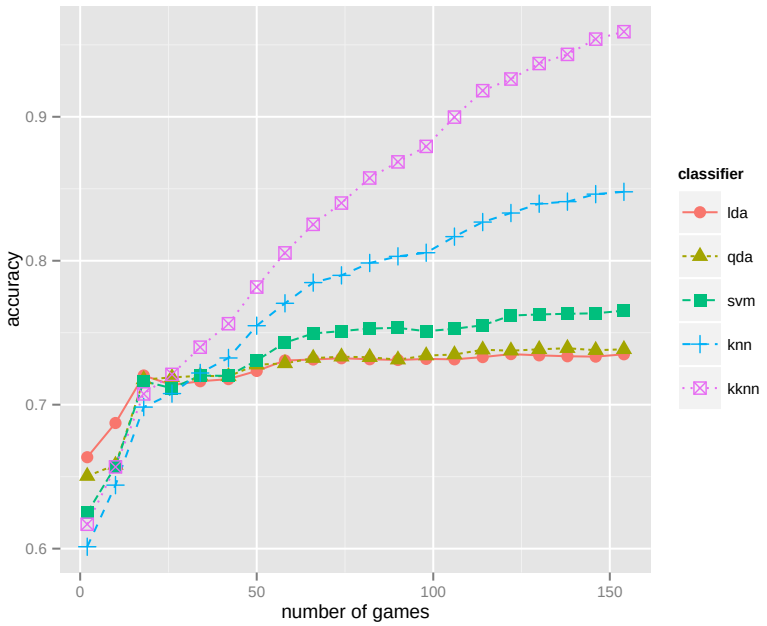


Fig. 4: Accuracy of the classifiers vs. the number of training games.

the formation of the armies (*pre-battle* vs. *during battle* prediction). LDA, QDA and SVM do not reach 90% of accuracy until the last quarter of the game. kNN is able to reach the same accuracy at 66% of the game. The results of kkNN are spectacular again, classifying correctly 90% of the game states from the first seconds. kkNN is the only algorithm able to effectively find useful patters in the training data before the armies begin to fight. Our intuition is that the training data is biased somehow, probably because the StarCraft AI is playing against itself and it does not use so many different attack strategies. In any case, kkNN seems to be the only algorithm to effectively predict the outcome of the battle from the first moves of each army.

Another important aspect when choosing a classifier is the volume of training data they need to perform well. Figure 4 shows the accuracy of each classifier as we increase the number of games used during the training phase. In the first 20 games all the algorithms perform similarly but then only kNN and kkNN keep improving fast. It makes sense for instance-based algorithms to require a large number of samples to achieve their highest degree of accuracy in complex domains, while algorithms that infer general models stabilize earlier but their prediction is more biased.

Finally, Figure 5 shows the stability of the predictions. We divided the game in 20 intervals of 5% of time. The y-axis represents the number of games for which the classifier made a prediction in that time interval that remained stable

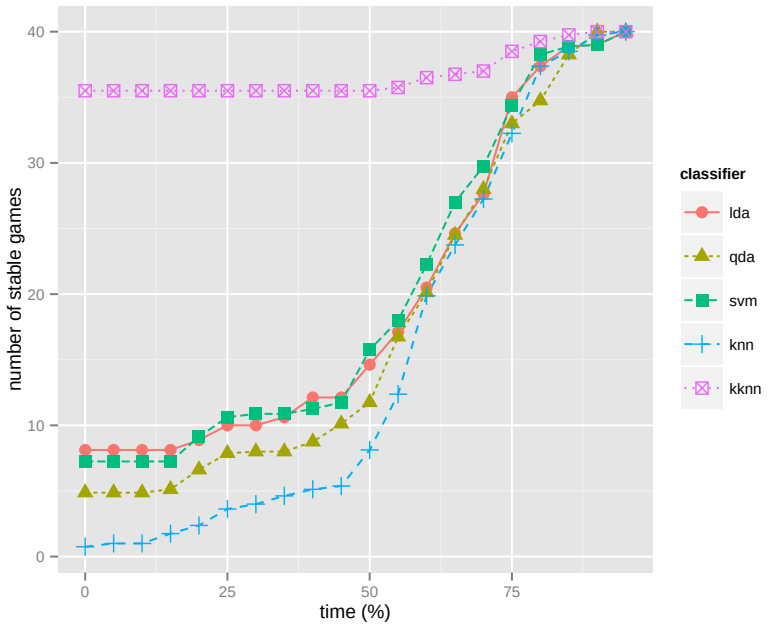


Fig. 5: Number of games for which each classifier becomes stable at a given time.

for the rest of the game. For example, the y value for $x = 0$ represents the number of games in which the prediction became stable during the first time interval (0-4.99% of the game). Most of the classifiers need to wait until the last quarter of the game to be stable in 80% of the games, except kkNN that is very stable from the beginning. There are a few games, however, in which all the classifiers are wrong until the end of the game because the army that was winning made bad decisions during the last seconds.

In conclusion, instance-based classifiers seems to perform better in our scenario, and kkNN in particular is the only algorithm that is able to effectively find useful patters in the training data before the armies begin to fight. It is also the most stable and it only performs worst than the other algorithms where there is a very small number of training games available.

6 Related work

RTS games have captured the attention of AI researchers as testbeds because they represent complex adversarial systems that can be divided into many interesting subproblems [4]. Proof of this are the different international competitions in AIIDE and CIG conferences. We recommend [12] for a complete overview of the existing work on this domain, the specific AI challenges and the solutions that have been explored so far.

There are several papers regarding the combat aspect of RTS games. [6] describes a fast Alpha-Beta search method that can defeat commonly used AI scripts in small combat scenarios. It also presents evidence that commonly used combat scripts are highly exploitable. A later paper [5] proposes new strategies to deal with large StarCraft combat scenarios.

Several different approaches have been used to model opponents in RTS games in order to predict the strategy of the opponents and then be able to respond accordingly: decision trees, kNN, logistic regression [17], case-based reasoning [1, 3], bayesian models [16] and evolutionary learning [13] among others.

A paper very related to our work is [15], where authors present a Bayesian model that can be used to predict the outcome of isolated battles, as well as to predict what units are needed to defeat a given army. Our approach is different in the sense that we try to predict the outcome as the game progresses and our battles begin with 2 balanced armies (same number and type of units). We use tactical information regarding the location of the troops and we use StarCraft to run the experiments and not a battle simulator.

7 Conclusions and Future work

In this paper we compare different machine learning algorithms in order to predict the outcome when two small marine armies engage in combat in the StarCraft game. The predictions are made from the perspective of an external game observer so they are based only on the actions of the individual units. The proposed approach is not limited to RTS games and can be used in other domains like multi-agent simulations, since it does not depend on whether the actions are decided by each unit autonomously or by a global manager. Our results indicate that, in this scenario, instance-based classifiers such as kNN and kkNN behave much better than other classifiers that try to infer a general domain model in terms of accuracy, size of the training set and stability.

There are several possible ways to extend our work. We have only considered a small battle scenario with a limited number of units. As the number and diversity of units increases, the number of possible combat strategies also grows creating more challenging problems. Our map is also quite simple and flat, while most of the StarCraft maps have obstacles, narrow corridors, wide open areas and different heights providing locations with different tactical value. The high accuracy values obtained by kkNN from the first seconds of the battle make us suspicious about the diversity of the strategies in the recorded games. We plan to run new experiments using human players to verify our results. Finally, our predictions are based on static pictures of the current game state. It is reasonable to think that we could improve the accuracy if we consider the evolution of the game and not just the current state to predict the outcome of the battle.

References

1. Aha, D.W., Molineaux, M., Ponsen, M.: Learning to win: Case-based plan selection in a real-time strategy game. In: in Proceedings of the Sixth International

Conference on Case-Based Reasoning. pp. 5–20. Springer (2005)
2. Altman, N.S.: An Introduction to Kernel and Nearest-Neighbor Nonparametric Regression. American Statistician 46, 175–185 (1992)
3. Auslander, B., Lee-Urban, S., Hogg, C., Muñoz-Avila, H.: Recognizing the enemy: Combining reinforcement learning with strategy selection using case-based reasoning. In: Advances in Case-Based Reasoning, 9th European Conference, ECCBR 2008, Trier, Germany, September 1-4, 2008. Proceedings. pp. 59–73 (2008)
4. Buro, M., Furtak, T.M.: RTS games and real-time AI research. In: In Proceedings of the Behavior Representation in Modeling and Simulation Conference (BRIMS. pp. 51–58 (2004)
5. Churchill, D., Buro, M.: Portfolio greedy search and simulation for large-scale combat in starcraft. In: 2013 IEEE Conference on Computational Inteligence in Games (CIG), Niagara Falls, ON, Canada, August 11-13, 2013. pp. 1–8 (2013)
6. Churchill, D., Saffidine, A., Buro, M.: Fast Heuristic Search for RTS Game Combat Scenarios. In: Proceedings of the Eighth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE-12, Stanford, California, October 8-12, 2012 (2012)
7. Cortes, C., Vapnik, V.: Support-Vector Networks. Mach. Learn. 20(3), 273–297 (Sep 1995)
8. Fisher, R.A.: The Use of Multiple Measurements in Taxonomic Problems. Annals of Eugenics 7(7), 179–188 (1936)
9. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning. Springer Series in Statistics, Springer New York Inc., New York, NY, USA (2001)
10. Hechenbichler, K., Schliep, K.: Weighted k-Nearest-Neighbor Techniques and Ordinal Classification (2004)
11. James, G., Witten, D., Hastie, T., Tibshirani, R.: An Introduction to Statistical Learning: with Applications in R. Springer Texts in Statistics, Springer (2013)
12. Ontañón, S., Synnaeve, G., Uriarte, A., Richoux, F., Churchill, D., Preuss, M.: A Survey of Real-Time Strategy Game AI Research and Competition in StarCraft. IEEE Trans. Comput. Intellig. and AI in Games 5(4), 293–311 (2013)
13. Ponsen, M.J.V., Muñoz-Avila, H., Spronck, P., Aha, D.W.: Automatically Acquiring Domain Knowledge For Adaptive Game AI Using Evolutionary Learning. In: Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA. pp. 1535–1540 (2005)
14. Samworth, R.J.: Optimal weighted nearest neighbour classifiers. Ann. Statist. 40(5), 2733–2763 (10 2012)
15. Stanescu, M., Hernandez, S.P., Erickson, G., Greiner, R., Buro, M.: Predicting Army Combat Outcomes in StarCraft. In: Proceedings of the Ninth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE-13, Boston, Massachusetts, USA, October 14-18, 2013 (2013)
16. Synnaeve, G., Bessière, P.: A Bayesian Model for Plan Recognition in RTS Games Applied to StarCraft. In: Proceedings of the Seventh AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, AIIDE 2011, October 10-14, 2011, Stanford, California, USA (2011)
17. Weber, B.G., Mateas, M.: A Data Mining Approach to Strategy Prediction. In: Proceedings of the 5th International Conference on Computational Intelligence and Games. pp. 140–147. CIG'09, IEEE Press, Piscataway, NJ, USA (2009)

Multi-Agent Case-Based Diagnosis in the Aircraft Domain

Pascal Reuss[1,2], Klaus-Dieter Althoff[1,2], Alexander Hundt[1], Wolfram Henkel[3], and Matthias Pfeiffer[3]

[1] German Research Center for Artificial Intelligence
Kaiserslautern, Germany
http://www.dfki.de
[2] Institute of Computer Science, Intelligent Information Systems Lab
University of Hildesheim, Hildesheim, Germany
http://www.uni-hildesheim.de
[3] Airbus
Kreetslag 10 21129 Hamburg, Germany

Abstract. Aircraft diagnosis is a highly complex topic. Many knowledge sources are required and have to be integrated into a diagnosis system. This paper describes the instantiation of a multi-agent system for case-based aircraft diagnosis based on the SEASALT architecture. This system will extend a existing rule-based diagnosis system, to make use of the experience of occurred faults and their solutions. We describe the agents within our diagnosis system, the planned diagnosis workflow and the current status of the implementation. For the case-based agents, we give an overview of the initial case structures and similarity measures. In addition, we describe some challenges we have during the development of the multi-agent system, especially during the knowledge modeling.

1 Introduction

An aircraft is a complex mechanism, consisting of many subsystems. Occurring faults cannot be easily tracked to their root cause. A fault can be caused by one system, by the interaction of several systems or by the communication line between systems. Finding the root cause can be very time and resource consuming. Therefore the use of experience from successfully found and solved root causes can be very helpful for aircraft diagnosis. This paper describes the instantiation of a multi-agent system (MAS) based on the SEASALT architecture. The MAS contains several Case-Based Reasoning (CBR) systems to store the experience and provide this knowledge for diagnosis. In the next section, we give an overview of the OMAHA project and the SEASALT architecture. Section 2 contains related work with comparing our approach to other diagnosis and multi-agent approaches. In Section 3 we describe the instantiation of the SEASALT components for our MAS and describe the case-bases agents with the case structure and similarity measures of the underlying CBR systems in more detail. Section 4 gives a short summary of the paper and an outlook on future work.

1.1 OMAHA project

The multi-agent system for case-based aircraft diagnosis is under development in the context of the OMAHA research project. This project is supported by the German Federal Ministry of Economy and Energy and tries to develop an **O**verall **M**anagement **A**rchitecture for **H**ealth **A**nalysis of civilian aircraft. Several topics are addressed within the project like diagnosis and prognosis of flight control systems, innovative maintenance concepts, and effective methods of data processing and transmission. A special challenge of the OMAHA project is to integrate not only the aircraft and its subsystems, but also systems and processes in the ground segment like manufacturers, maintenance facilities, and service partners into the maintenance process. Several enterprises and academic and industrial research institutes take part in the OMAHA project: the aircraft manufacturer Airbus, the system manufacturers Diehl Aerospace and Nord-Micro, the aviation software solutions provider Linova and IT service provider Lufthansa Industrial Solutions as well as the German Research Center for Artificial Intelligence and the German Center for Aviation and Space. In addition, several universities are included as subcontractors. The project started in 2014 and will last until the end of March, 2017. [1]

The OMAHA project has several different sub-projects. Our work focuses on a sub-project to develop a so-called integrated system health monitoring (ISHM) for aircraft systems. The main goal is to improve the existing diagnostic approach to identify faults with root cause in more than a single subsystem (cross-system faults). Therefore, a multi-agent system (MAS) with several case-based agents will be implemented to integrate experience into the diagnostic process and provide more precise diagnoses for given faults.

1.2 SEASALT architecture

The SEASALT (**S**hared **E**xperience using an **A**gent-based **S**ystem **A**rchitecture **L**ayou**T**) architecture is a domain-independent architecture for extracting, analyzing, sharing, and providing experiences [4]. The architecture is based on the Collaborative Multi-Expert-System approach [1],[2] and combines several software engineering and artificial intelligence technologies to identify relevant information, process the experience and provide them via an interface. The knowledge modularization allows the compilation of comprehensive solutions and offers the ability of reusing partial case information in form of snippets.

The SEASALT architecture consists of five components: knowledge sources, knowledge formalization, knowledge provision, knowledge representation, and individualized knowledge. The knowledge sources component is responsible for extracting knowledge from external knowledge sources like databases or web pages and especially Web 2.0 platforms.

The knowledge formalization component is responsible for formalizing the extracted knowledge into a modular, structural representation. This formalization

[1] `www.dlr.de/lk/desktopdefault.aspx/tabid-4447/7274_read-39606`

is done by a knowledge engineer with the help of a so-called Apprentice Agent. This agent is trained by the knowledge engineer and can reduce the workload for the knowledge engineer.

The knowledge provision component contains the so-called Knowledge Line. The basic idea is a modularization of knowledge analogous to the modularization of software in product lines. The modularization is done among the individual topics that are represented within the knowledge domain. In this component a Coordination Agent is responsible for dividing a given query into several sub queries and pass them to the according Topic Agents. The agent combines the individual solutions to an overall solution, which is presented to the user. The Topic Agents can be any kind of information system or service. If a Topic Agent has a CBR system as knowledge source, the SEASALT architecture provides a Case Factory for the individual case maintenance [4],[3],[9].

The knowledge representation component contains the underlying knowledge models of the different agents and knowledge sources. The synchronization and matching of the individualized knowledge models improves the knowledge maintenance and the interoperability between the components. The individualized knowledge component contains the web-based user interfaces to enter a query and present the solution to the user.

1.3 Application domain: aircraft diagnosis

An aircraft is a highly complex machine consisting of a large number of subsystems that interact with each other, like hydraulic, cabin, ventilation, and landing gear. Each subsystem has a large number of components. The smallest component that can be replaced during maintenance is called Line Replacement Unit (LRU). The challenge is to find the root cause of a fault, because there could be more than one LRU causing the fault or a fault chain. In a fault chain, the first fault causes additional faults, which could also cause additional faults again. Faults are not limited to have their root cause in the subsystems that stated the fault, but the root cause can be found in a different subsystem. Therefore, a cross-system diagnosis is required to improve the precision of the diagnosis process.

In the next section we give an overview of some related work. In Section 3 we describe the multi-agent system concept and the instantiation of the SEASALT architecture. Section 3.3 describes the current status of our implementation. Finally a summary and outlook on future work is given.

2 Related Work

Decision support for diagnosis (and maintenance) in the aircraft domain means that a lot of engineering knowledge is available to support this process. In the past various diagnostic approaches tried to improve diagnosis and maintenance in this domain: among others case-based reasoning, rule-based reasoning, model-based reasoning, Bayesian belief networks, Fuzzy inference, neural networks,

fault trees, trend analysis, and a lot of combinations. For OMAHA, that is OMAHA work package 230, the exploitation of available experiences as supplementation to other already used knowledge sources is of high priority. See also the work from Reuss et al.[10] for relating our approach with a selection of related other experience reusing diagnostic approaches: the British research project DAME [7] dealing with fault diagnosis and prognosis based on grid computing , Dynamic Case-Based Reasoning [13] learning also through statistic vectors containing abstract knowledge condensed from groups of similar cases, and the hybrid approach of Ferret and Glasgow [6] combining model-based and case-based reasoning.

For optimizing the relation between cost and benefit we decided to use the various available textual knowledge sources (cf. also Section 3). A recent overview of using textual sources for CBR is given in the textbook of Richter and Weber [12]. The paper of Reuss et al. [11] also gives an overview of some related approaches in this direction.

In addition to other specific characteristics of our approach one property differentiating it from many other (CBR) approaches is the fact that we develop a multi-agent system that applies a lot of CBR agents (among other) ones. The following approaches have in common that they also combine a multi-agent system approach with CBR. Researchers also dealing with CBR from different perspectives and trying to combine the specific insights to an improved overall approach are [16]. Of course, what makes our approach different here is that we are concerned with the development of concrete framework with existing applications. Corchado et al.[5] present in their work an architecture for integrating multi-agent systems, distributed services, and application for constructing Ambient Intelligence environments. Besides addressing a different domain and task this approach appears to be more open concerning the potential tasks agents can take over, while our approach is more focused in applying software engineering strategies for decomposing problems into sub-problems resulting in a distributed knowledge-based system. Zouhaire and his colleagues[17] developed a multi-agent system using dynamic case-based reasoning that learns from traces and is applied for (intelligent) tutoring. Our approach does not learn from traces but instead has to deal with a lot of technical knowledge and in addition has to solve critical problems. Srinivasan, Singh and Kumar[14] share with our approach that they develop a conceptual framework for decision support systems based on multi-agent systems and CBR systems. Our approach appears to be more on the side of integrating software engineering and artificial intelligence methods implementing concrete application systems, while the authors discuss how their framework influences decision support system in general. Khelassi[8] developed the IK-DCBRC system basing on a multi-agent architecture using a CBR approach with fuzzy-enhanced similarity assessment and being able to explain the results for different users. Our approach is not explanation-aware with respect to its current implementation status, however there is a conceptional extension of the SEASALT architecture (together with Thomas Roth-Berghofer and his research team) defined that includes explanation awareness. In addition,

there are two PhD research projects ongoing focusing on explanation awareness. What also makes us different from Khelassis work is that our approach is embedded in an overall methodology resulting in a systematic process of how to develop an instance of our architecture with applications in travel medicine, technical diagnosis, and architectural design.

3 Multi-agent case-based diagnosis in the aircraft domain

In this section we describe the current version of our multi-agent system for case-based diagnosis. Based on the SEASALT architecture we describe the instantiation of the single components in context of our multi-agent system and the diagnosis workflow. In addition, we give an overview over the case structures and similarity measures of our case-based agents.

3.1 Multi-agent system for aircraft diagnosis

First we will describe the instantiation of our multi-agent system. The multi-agent system is an additional component of the diagnosis mechanism. It will not replace the existing rule-based diagnosis, but will extend the current diagnosis mechanism. The main component for our multi-agent aircraft diagnosis is the *knowledge provision* component. This component contains the Knowledge Line, which is responsible for providing a diagnosis for a given fault situation. The Knowledge Line consists of several topic agents with underlying CBR systems. The topic agents use the knowledge of their CBR systems to provide a part of the diagnosis. If only the knowledge of one topic agent is required, the topic agents delivers the complete diagnosis. There are several homogeneous teams of topic agents in the Knowledge Line, each responsible for diagnoses of an aircraft type (e.g., A320, A350, or A380). Each team has an additional agent, called solution agent to coordinate the topic agents and rank the individual solutions. Because each individual solution represents a possible diagnosis, a combination of solutions is not appropriate. The approach of separated agent teams for each aircraft type is based on the idea to split the knowledge into several smaller CBR systems. This way the number of cases for a retrieval and the maintenance effort for each system can be reduced. Nevertheless, for a diagnosis more than one agent team may be necessary. Therefore, a query can be distributed to several agent teams, either by default or if a diagnosis from the primary agent team for a query cannot provide a sufficient diagnosis. A coordination agent is responsible for coordinating the agent teams, distributing a query, and combining the team's solutions. The complete diagnosis process requires some more software agents that do not belong to the Knowledge Line itself: an interface agent, a composition agent, a knowledge map agent, and an output agent. The interface agent receives the query either from a web interface and/or a data warehouse. The main data source is a Post Flight Report (PFR) containing all the faults having occurred during the last flight of an aircraft. This PFR is based on the rule-based diagnosis in the aircraft. Each fault is represented as a so-called PFR

item. Additional data like aircraft configuration, operational parameters (e.g., weather conditions, temperature, etc.), and logbook entries can be received, too. The PFR data and the additional data have to be correlated to assign the additional data to the corresponding PFR item. This task is done by the correlation agent. The extended PFR items are sent to the coordination agent. For each PFR item, a request to one or more agent teams is performed. To determine which topic agents of a team should be requested, a so-called Knowledge Map is used. This Knowledge Map contains information about existing agents and their dependencies and underlying CBR systems. The task to determine a so-called retrieval path (the topic agents to be requested and the sequence of retrievals) is done by a knowledge map agent. This agent has access to the general Knowledge Map and a CBR system, which stores past successful retrieval paths for given fault situations. The knowledge map agent uses the CBR system to retrieve the most similar retrieval paths and adapt the path to the new situation if necessary. Based on the determined retrieval path, the topic agents are requested and a ranked list of diagnoses is generated. The list of diagnoses is sent to the output agent. The output agent forwards the list to the web interface and the data warehouse. One more agent is located in the *knowledge provision* component. The so-called query analyzer takes each extended PFR item and checks for new concepts, which are not yet part of the vocabulary of the CBR systems. If any new concepts are found, a maintenance request is sent to the so-called Case Factory [9]. The Case Factory checks the maintenance request, derives appropriate maintenance actions, and executes the actions after confirmation from a knowledge engineer. The query analyzer is not part of the diagnosis process itself, but provides some learning capabilities to the multi-agent system.

The user interface can be found in the *individualized knowledge* component. The user interface is a web interface, which provides the options to send a query to the multi-agent system and present the returned diagnoses. In addition, the user can enter new cases, edit existing cases, and browse a entire selected case base. In addition to the web interface, a connection to a data warehouse is part of this component. The data warehouse contains PFRs and the additional data and will be the main query source for the multi-agent systems. If additional information is required that is not provided by the data warehouse, it can be added via the web interface.

The *knowledge formalization* component transforms structured, semi structured, and unstructured data into a modular, structural knowledge representation used by all CBR systems. This way the knowledge is represented in the same way all over the multi-agent system. Because a structural approach for the CBR systems in the Knowledge Line was chosen, semi-structured and unstructured data have to be transformed into attribute value pairs. This transformation workflow is performed by a so-called case base input analyzer. The workflow consists of several steps: At first, information extraction methods are used to extract keywords and collocations and to find synonyms and hypernyms for the extracted keywords. Then the input data is analyzed to find associations within the allowed values of an attribute as well as across different attributes.

This way want to generate completion rules for query expansion. The keywords, synonyms, hypernyms, and collocations are added to the vocabulary and initial similarity values for keywords and their synonyms are set. The keywords and their hypernyms can be used to generate taxonomies for similarity measures. After the vocabulary extension, cases are generated from the input data and stored in the case bases. The last step is to perform a sensitivity analysis on the stored cases to determine the weighting for the problem description attributes. The workflow is presented in more detail in [11].

In the *knowledge sources* component a collector agent is responsible for finding new data in the data warehouse, via web services or in existing knowledge sources of Airbus. New data could be new configurations or operational parameters, new synonyms or hypernyms, or complete new cases.

The *knowledge representation* component contains the generated vocabulary, similarity measures and taxonomies, completion rules, and constraints provided for all agents and CBR systems.

3.2 Case-based agents

This section focuses on the case-based agents within our multi-agent diagnosis system. We will describe the agents' tasks and the underlying CBR systems with their case structure and similarity modeling. In addition to the PFR, we have to consider several different data structures like Service Information Letters (SIL), In-Service Reports (ISR), elogbooks and aircraft configuration documents. While a PFR contains only information about the problem description, SIL, ISR and eLogbooks contain problem descriptions and solutions. Configuration documents contain data about the latest system configuration of an aircraft with hard- and software versions. We performed an analysis on these data to identify relevant information for cases, relationships between these information and data anomalies. Based on the result of this analysis we derived two case structures with attribute-value pairs and their value ranges. One case structure is based on PFR and SIL (C_{SIL}) and the other case structure is based on PFR and ISR (C_{ISR}). The case structures overlap to some degree, because attributes derived from the PFRs are part of both structures, like ATA chapter, aircraft type, and fault description. The C_{SIL} structure contains 32 attributes, while the C_{ISR} structure consists of 28 attributes. The attributes are distributed among problem description, diagnosis, quality information, and pointer to other cases. The problem description contains attributes like ATA chapter, aircraft type (e.g., A380), aircraft model (e.g., 380-641), fault code, displayed message, fault description and affected Line Replacement Units (LRU). Attributes like recommendation, comments, maintenance reference, corrective LRUs and root cause are part of the solution. For quality assessments the number of positive (a retrieved diagnosis was helpful) and negative (a retrieved diagnosis was not helpful) retrievals are stored.

The configuration of an aircraft has great impact on the probability of fault occurrence. If a certain system is not built in, corresponding faults will not occur. The occurrence of faults depends also on the soft- and hardware version

of built in systems. Therefore, the configuration of an aircraft can exclude faults and root causes and have an impact on the similarity of cases. Because of the complexity of the configuration data for an entire aircraft, we decided to consider the configuration separate for each aircraft component. For each subsystem of a component the so-called modification status (mod-status) is stored. With the help of this mod-stati, cases could be excluded and similar configuration could be compared.

Most of the attributes have a symbolic data type and a taxonomy as similarity measure. The attributes ATA chapter, fault code and affected LRUs have a natural hierarchical structure, that can be mapped to a taxonomy. A great challenge is the similarity measure for the fault description attribute. The symbolic values of this attribute are extracted via a workflow in the knowledge formalization component as described in [11]. During the automatic vocabulary expansion, the values are added to a similarity table. Similarities between the automatically added values are only set between values and their synonyms. The other values have to be set manually. To reduce the effort, an automatic taxonomy generation from the extracted values and their synonyms and hypernyms is planned.

The multi-agent system will contain several topic agents with the same case structure to reduce the number of cases in one case base. Most faults can be assigned to a specific ATA chapter. Therefore, for each ATA chapter an own topic agent is generated. An agent team within our multi-agent system will consist of agents discriminated by ATA chapter and data source (SIL, ISR, etc.).

Another case-based agent is the so-called knowledge map agent. This agent is responsible for determining which topic agents have to be requested to find a solution for a given request. For each request, a retrieval on the underlying CBR system is performed. The cases will contain the characteristics of a request as the problem description and a successfully used retrieval path. This way we try to address the cross-system faults. Cross-system faults may have their root causes in LRUs of different ATA chapters. Requesting only the topic agent of a single ATA chapter may not give the correct root cause identification and diagnosis. Based on experience from solved faults, the cases for the knowledge map agent could contain information when the request of additional topic agents may be useful to find the correct diagnosis.

There are several challenges to be met while modeling the case structures and the similarity measures. One major challenge is based on the fact, that the ATA chapter differs for the same subsystem in different aircraft types. The cabin entertainment system is linked to two different ATA chapters in the A320 and the A380. Therefore, a mapping between the different ATA chapters is required to compare fault cases from different aircraft types. Another challenge is modeling the fault description in the case structure. The description of a fault is mainly given in free text provided by pilots or cabin personal. Unfortunately, there is no standard description language for faults. Therefore, every person describes a fault with slightly different words and technical terms. Extracting the key symptoms from this fault descriptions and comparing two fault descriptions requires the integration of natural language processing techniques in the modeling process

and the diagnosis process of the multi-agent system. In addition, the amount of knowledge that can be found in the fault descriptions is very high. Analyzing 3000 example fault descriptions, we found more than 21000 different keywords and phrases describing the occurred faults. Modeling all these keywords and phrases in one attribute is not practicable. While it is possible to add all keywords automatically, setting the similarity between these keywords within a similarity matrix or a taxonomy is not practicable. In addition, the maintenance effort for such an attribute would be very high and in no relation the gained benefit.

The main challenge for the knowledge map agent is to identify the characteristics of a request and the according knowledge sources to solve the request.

3.3 Status of implementation

We implemented a prototype to test some functionalities of the desired multi-agent system. This application serves as a testing system for knowledge modeling and diagnosis process. The prototype consists of two CBR systems and a user interface to interact with the systems. We modeled the case structure, vocabulary and similarity using the open source tool myCBR [15]. One CBR system contains cases based on SIL documents, the other one on ISR documents. The SIL case base contains 670 cases and the ISR case base 220 cases. The user interface provides the functionalities to perform a retrieval, enter new cases, edit existing cases, and browse the case base based on filter criteria. In addition, the workflow of the knowledge extraction is partially implemented. The keyword extraction, collocation extraction, synonym and hypernym identification, and automatic vocabulary extension are implemented. For more detail on the implementation of the knowledge extraction workflow see [11].

4 Summary and Outlook

In this paper we describe the instantiation of our multi-agent system for case-based diagnosis. We give an overview of the individual components and describe the case structure and similarity of our case-based agents. As Section 3.3 shows, the multi-agent system is not fully implemented, yet. The next steps are the implementation of the additional agents (interface, coordination, output, knowledge map) and the refinement of the case structures and similarity measures. In addition, the learning mechanism based on the knowledge extraction workflow will be realized.

References

1. Althoff, K.D.: Collaborative multi-expert-systems. In: Proceedings of the 16th UK Workshop on Case-Based Reasoning (UKCBR-2012), located at SGAI International Conference on Artificial Intelligence, December 13, Cambride, United Kingdom. pp. 1–1 (2012)

2. Althoff, K.D., Bach, K., Deutsch, J.O., Hanft, A., Mänz, J., Müller, T., Newo, R., Reichle, M., Schaaf, M., Weis, K.H.: Collaborative multi-expert-systems – realizing knowledge-product-lines with case factories and distributed learning systems. In: Baumeister, J., Seipel, D. (eds.) KESE @ KI 2007. Osnabrück (Sep 2007)
3. Althoff, K.D., Hanft, A., Schaaf, M.: Case factory - maintaining experience to learn. Advances in Case-Based Reasoning Lecture Notes in Computer Science 4106/2006, 429–442 (2006)
4. Bach, K.: Knowledge Acquisition for Case-Based Reasoning Systems. Ph.D. thesis, University of Hildesheim (2013), dr. Hut Verlag Mnchen
5. Corchado, J.M., Tapia, D.I., Bajo, J.: A multi-agent architecture for distributed services and applications. International Journal of Innovate Computing 8, 2453–2476 (2012)
6. Feret, M., Glasgow, J.: Combining case-based and model-based reasoning for the diagnosis of complex devices. Applied Intelligence 7, 57–78 (1997)
7. Jackson, T., Austin, J., Fletcher, M., Jessop, M.: Delivering a grid enabled distributed aircraft maintenance environment (dame). Tech. rep., University of York (2003)
8. Khelassi, A.: Reasoning System for Computer Aided Daignosis with explanation aware computing for medical applications. Ph.D. thesis, Abou Bakre Belkaied University, Tlemcen, Algeria (2013)
9. Reuss, P., Althoff, K.D.: Explanation-aware maintenance of distributed case-based reasoning systems. In: LWA 2013. Learning, Knowledge, Adaptation. Workshop Proceedings. pp. 231–325 (2013)
10. Reuss, P., Althoff, K.D., Henkel, W., Pfeiffer, M.: Case-based agents within the omaha project. In: Case-based Agents. ICCBR Workshop on Case-based Agents (ICCBR-CBR-14) (2014)
11. Reuss, P., Althoff, K.D., Henkel, W., Pfeiffer, M., Hankel, O., Pick, R.: Semi-automatic knowledge extraction from semi-structured and unstructured data within the omaha project. In: Proceedings of the 23rd International Conference on Case-Based Reasoning (2015)
12. Richter, M.M., Weber, R.: Case-Based Reasoning - A Textbook. Springer-Verlag Berlin Heidelberg (2002)
13. Saxena, A., Wu, B., Vachtsevanos, G.: Integrated diagnosis and prognosis architecture for fleet vehicles using dynamic case-based reasoning. In: Autotestcon 2005 (2005)
14. Srinivasan, S., Singh, J., Kumar, V.: Multi-agent based decision support system using data mining and case based reasoning. International Journal of Computer Science Issues 8, 340–349 (2011)
15. Stahl, A., Roth-Berghofer, T.: Rapid prototyping of cbr applications with the open source tool mycbr. In: Advance in Case-Based Reasoning, Proceeding of the 9th European Conference on Case-Based Reasoning (2008)
16. Sun, Z., Han, J., Dong, D.: Five perspectives on case based reasoning. In: 4th International Conference on Intelligent Computing. pp. 410–419 (2008)
17. Zouhair, A., En-Naimi, E.M., Amami, B., Boukachour, H., Person, P., Bertelle, C.: Incremental dynamic case based reasoning and multi-agent systems (idcbr-mas) for intelligent touring system. International Journal of Advanced Research in Computer Science and Software Engineering 3, 48–56 (2013)

A Case-Based Framework for Task Demonstration Storage and Adaptation

Tesca Fitzgerald, Ashok Goel

School of Interactive Computing,
Georgia Institute of Technology,
Atlanta, Georgia, 30332
{tesca.fitzgerald,goel}@cc.gatech.edu

Abstract. We address the problem of imitation learning in interactive robots which learn from task demonstrations. Many current approaches to interactive robot learning are performed over a set of demonstrations, where the robot observes several demonstrations of the same task and then creates a generalized model. In contrast, we aim to enable a robot to learn from individual demonstrations, each of which are stored in the robot's memory as source cases. When the robot is later tasked with repeating a task in a new environment containing a different set of objects, features, or a new object configuration, the robot would then use a case-based reasoning framework to retrieve, adapt, and execute the source case demonstration in the new environment. We describe our ongoing work to implement this case-based framework for imitation learning in robotic agents.

Keywords: Case-based agents, imitation learning, robotics

1 Introduction

Imitation is an essential process in human social learning and cognition [11, 10]. Imitation learning occurs when a learner observes a teacher demonstrating some action, providing knowledge of (i) how the action was performed and (ii) the resulting effects of that action. This interaction-guided learning method allows us to learn quickly and effectively. As a result of its importance in human cognition, it follows that imitation learning has become an area of increasing focus in interactive robotics research as well.

The goal of *Learning from Demonstration* is to enable imitation learning in robots through interactive demonstrations, provided through methods such as teleoperation or kinesthetic teaching [1, 2]. Regardless of which demonstration method is used, the following process is often used. First, the human teacher provides several demonstrations of a skill. Between demonstrations, the teacher may adjust the environment such that the skill is demonstrated in a variety of initial configurations. The robot then creates an action model which generalizes over the provided demonstrations. Lastly, the robot applies the generalized action model to plan a trajectory which is executed in a new environment.

However, a challenge of this process is that the resulting action model is dependent on the number of demonstrations that were provided for that particular task. We also assume that the robot has been exposed to enough variations of the initial configuration such that its generalized model can be applied to a wide range of related initial configurations. As such, the generalized model is restricted to application in environments which are similar to those demonstrated.

We describe our preliminary work toward defining an alternate approach to imitation learning in robotics, one which takes a *case-based* approach in which the robot stores demonstrations *individually* in memory. We define a case-based framework which enables the full imitation learning process, from observing a task demonstration to transfer and execution. We also define a case representation which encodes task demonstrations for storage in source case memory.

2 Related Work

Case-based reasoning has been used to address the problem of transfer in robotics domains. Floyd, Esfandiari & Lam [7] describe a CBR approach to learning strategies for RoboCup soccer by observing spatially distributed soccer team plays. Their approach represents each case as an encoding of a single agent's perception and resulting action at a given time. Thus, they transfer the behavior of an agent when it perceives a situation similar to that of the observed agent. More recently, Floyd & Esfandiari [6] describe an approach for case-based learning by observation in which strategy-level domain-independent knowledge is separated from low-level, domain-dependent information such as the sensors and effectors on a physical robot. Ontañón et al. [8] describe their approach to observational learning for agents in real-time strategy games. They use a case-based approach to online planning, in which agents adapt action plans which are observed from game logs of expert demonstrations.

While these approaches do address knowledge transfer for robotic and simulated agents, they primarily operate over input and output represented at a higher level of abstraction, such as actions at a strategic level. The goal of our work is to enable transfer to generate action at a lower level of control and in response to real-world perceptual input, where we transfer the demonstrated action trajectory used to achieve a task. We expand on our previous work [3] describing a case-based approach to interpretation and imitation in robotic agents. We discussed two separate processes: (i) interpreting new skill demonstrations by comparing it to previously observed demonstrations using a case based process (further described in [5]), and (ii) a related process for imitating a task demonstration. This paper expands on the latter process, case-based imitation.

We previously provided a general outline for imitation in [3] in which four steps occur: representation of the task demonstration at multiple levels of abstraction, retrieval of the most relevant source case from memory, adaptation of the source case to address the target problem, and execution of the adapted case in the target problem. In this paper, we describe our more recent work providing (i) a revised, complete process of imitation beginning with observation

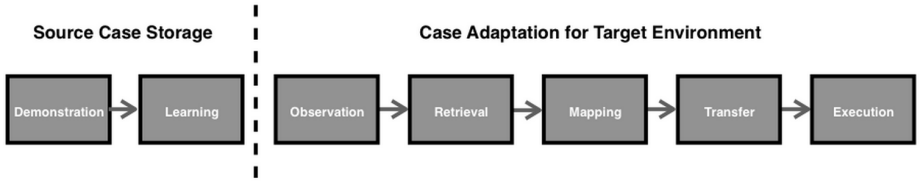


Fig. 1. Case-Based Process for Task Demonstration Transfer

of the task demonstration and ending with task transfer and execution, (ii) a *Mapping* step which bridges the gap between the *Retrieval* and *Transfer* steps, and (iii) a revised case representation for storing task demonstrations (iterating on preliminary work introduced in [4]).

3 Approach Overview

We have revised our case-based approach to transfer (originally summarized in [3]) to consist of two separate processes, as shown in Figure 1: the *Case Storage* process in which the robot receives demonstrations of a task and stores each demonstration as a case in source memory, and a *Case Adaptation* process which is used at a later time when the robot is asked to repeat a task in a target environment.

3.1 Why a CBR approach?

Our eventual goal is to enable transfer for imitation learning in scenarios such as the following. A human teacher guides the robot to complete a task such as scooping the contents of one container into another. During the demonstration, the robot records the demonstrated trajectories and object features. At a later time, the robot is asked to repeat the *scooping* task, but in a new, *target* environment. Thus, the robot must use a different set of object features to parameterize and execute the *scooping* task than those observed in the original, *source* environment. Next, the robot transfers its representation of the *scooping* task to accommodate for the differences between the source and target environments. The transferred task representation is then executed in the target environment.

Rather than generalize over a set of demonstrations as in current Learning from Demonstration methods (surveyed in [1, 2]), using a case-based approach allows us to: (1) operate under the assumption that the human teacher will provide a limited number of demonstrations, (2) represent demonstrations as individual experiences in the robot's memory, and (3) utilize a complete framework for transferring skill demonstrations, which includes the steps of retrieving, analyzing, transferring, and executing a relevant *source case* demonstration in an unfamiliar, *target environment*.

3.2 Case Storage Process

Demonstration and Learning We have implemented the first step in the *Case Storage* process, where the robot records and stores each task demon-

stration as a source case in memory. We define each case as the tuple $C = <L,\ D,\ T,\ O,\ S_i,\ S_f>$, where:

- L represents the label of the task which was demonstrated, e.g. "scooping".
- D represents the set of action models which encode the demonstrated motion, represented as Dynamic Movement Primitives as defined in [9].
- T is the set of parameterization functions which relate the set of action models to the locations of objects in the robot's environment. For example, a parameterization function may be used to represent how the robot's hand must be located above a bowl prior to completing a *pouring* action.
- O is the set of *salient* object IDs which are relevant to the task.
- S_i and S_f are the initial and final states, respectively, which represent the set of objects observed in an overhead view of the robot's environment.

3.3 Case Adaptation Process

At a later time, the robot may be asked to repeat the task in a new, target environment. We are currently implementing the Case Adaptation process shown in Figure 1.

Observation will begin when the robot is asked to address a target problem. We assume that the robot has been provided a relevant source case which it can retrieve from memory to address the given target problem. The robot will then observe the target environment by viewing the objects located in the table-top environment using an overhead camera. This will provide it with the target case's initial state S_i.

Retrieval must be performed to select a source case from memory containing the demonstration that is most relevant to the current target problem. Case retrieval will prioritize (i) similarity of task goals, (ii) similarity of salient objects, and finally, (iii) similarity of initial states. Once a relevant source case has been retrieved, the *Mapping* step must encode the differences between the source and target environments. This mapping will be later used to transfer the source case such that differences in the target environment are addressed.

Given a source case and mapping which encodes the differences between the source and target cases, the *Transfer* step adapts the source case. We take a similarity-based approach to transfer, where we consider the similarity between the source case and target environments when defining transfer processes. As we encounter transfer problems in which the source and target problems become less similar, the source case is transferred at a different level of abstraction, such that only high-level features of that case are transferred. The adapted case is then *executed* in the target environment.

We have implemented three methods which implement the *Transfer* step, each of which operates by transferring the source case at a different level of abstraction. Once the source case has been transferred, it is used to plan and execute a new action trajectory. In preliminary experiments, we have evaluated each method separately such that we selected the level of abstraction at which transfer occurred in each target problem. These experiments have shown us that

by changing the level of abstraction at which a case is transferred, a robot can use a single source demonstration to address target environments of varying similarity to the source environment.

4 Future Work

We have implemented the Case Storage process and the last two steps of the Case Adaptation process, the *Transfer* and *Execution* steps. Currently, we manually provide the robot with the most relevant source case demonstration and a mapping between objects in the source and target environments. Thus, our next steps are to identify a method for autonomously determining this object mapping. Furthermore, future work will involve defining a process for identifying and retrieving an appropriate source case demonstration that is most applicable to a given transfer problem.

Acknowledgments

This work was supported by NSF Graduate Research Fellowship DGE-1148903.

References

1. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robotics and Autonomous Systems 57(5), 469–483 (2009)
2. Chernova, S., Thomaz, A.L.: Robot learning from human teachers. Synthesis Lectures on Artificial Intelligence and Machine Learning 8(3), 1–121 (2014)
3. Fitzgerald, T., Goel, A.: A case-based approach to imitation learning in robotic agents. Intl. Conf. on Case-Based Reasoning Workshop on Case-Based Agents (2014)
4. Fitzgerald, T., Goel, A.K., Thomaz, A.L.: Representing skill demonstrations for adaptation and transfer. AAAI Symposium on Knowledge, Skill, and Behavior Transfer in Autonomous Robots (2014)
5. Fitzgerald, T., McGreggor, K., Akgun, B., Thomaz, A.L., Goel, A.K.: Visual case retrieval for interpreting skill demonstrations. International Conference on Case-Based Reasoning (2015)
6. Floyd, M.W., Esfandiari, B.: A case-based reasoning framework for developing agents using learning by observation. In: 2011 23rd IEEE International Conference on Tools with Artificial Intelligence (ICTAI). pp. 531–538. IEEE (2011)
7. Floyd, M.W., Esfandiari, B., Lam, K.: A case-based reasoning approach to imitating robocup players. In: FLAIRS Conference. pp. 251–256 (2008)
8. Ontañón, S., Mishra, K., Sugandh, N., Ram, A.: Case-based planning and execution for real-time strategy games. In: Case-Based Reasoning Research and Development, pp. 164–178. Springer (2007)
9. Pastor, P., Hoffmann, H., Asfour, T., Schaal, S.: Learning and generalization of motor skills by learning from demonstration. In: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. pp. 763–768. IEEE (2009)
10. Piaget, J., Cook, M.T.: The origins of intelligence in children. (1952)
11. Tomasello, M., Kruger, A.C., Ratner, H.H.: Cultural learning. Behavioral and brain sciences 16(03), 495–511 (1993)

Case Based Disruption Monitoring

Joe Kann, Matthew Molineaux and Bryan Auslander

Knexus Research Corp.
174 Waterfront Street, Suite 310, National Harbor, MD 20745

firstname.lastname@knexusresearch.com

Abstract. Mine Countermeasures Missions (MCM) take place in very complex and uncertain environments which poses complexity for planning and explanation algorithms. In order to keep a mission on target, constant disruption monitoring and frequent schedule adjustments are needed. To address this capability gap, we have developed the Case-Based Disruption Monitoring and Analyzing (CDMA) algorithm. The CDMA algorithm automatically detects disruptions within a mission and attempts to determine possible root causes. Once confirmed, our second developed algorithm, CLOSR modifies existing schedules to compensate for these root causes. Evaluation of CDMA on simulated MCM operations demonstrates the effectiveness of case-based disruption monitoring. Both the CDMA and CLOSR algorithms, along with simulator, are enclosed with our KRePE system.

1 Introduction

Unforeseen disruptions occur when planning in the real world. When monitoring for such disruptions and providing an explanation as to why the disruption occurs, better insight is provided in order to fix the plan. Mine Countermeasure Missions (MCM) for example, uses planning constantly. MCM planning uses a variety of resources and each resource has its own set of capabilities and operational constraints, as well as characteristic failure points.

Mine Countermeasure Missions (MCM) must respond to frequent disruptions, and recovering from these disruptions can be complex. MCM missions involve the location, identification, and neutralization of enemy explosive ordnance in a maritime context. This is key to naval power projection and sea control, two core capabilities of U.S. maritime power, as characterized by *A Cooperative Strategy for 21st Century Seapower* [4]. Due to high complexity and uncertainty when scheduling MCM missions, accurate plans must be created and frequently revised once a mission has started. Frequent disruptions in MCM operations can occur due to many types such as: changes in sea state, visibility, weather, equipment failure, etc. Situations like these interfere with resource availability and/or readiness. Therefore, schedules for MCM operations require frequent changes and updates where the disruptions are monitored in order to keep the success of the mission. Current practice calls for manually observing all incoming data

for detection of issues that could cause a mission to fail. The manual process of monitoring for disruptions can be tedious and prone to error.

To meet this need, we are developing a system for MCM operation decision making and planning support called KRePE. KRePE builds upon a foundation of cognitive architecture components, algorithms and simulations. Housed within the KRePE architecture the Case-Based Disruption Monitoring and Analyzing (CDMA) algorithm performs monitoring and analysis of disruptions and Case-Based Local Schedule Repair (CLOSR) reschedules tasks that MCM planner operators perform on a frequent basis. Both the CDMA and CLOSR algorithms fall in a problem solving paradigm known as Case-based reasoning (CBR) by relying on general and specific knowledge of MCM operations, how operations might be disrupted, and how to fix these interruptions.

In this paper, we discuss the challenges of continuous situation monitoring, and root cause analysis of mission disruptions through case-based reasoning. We close with an empirical study that demonstrates this effective anomaly detection in order to generate schedule modifications that achieve mission success.

2 Mine Countermeasures Mission Scheduling & Operations

MCM operations involve the location, identification, and neutralization of sea mines [5]. These operations employ surface vehicles, aircraft, divers, and unmanned surface and underwater vehicles, and can take weeks to plan and execute. While the operations are taking place, they are disrupted early and often by events such as unforeseen weather conditions, technological failures, and incorrect enemy course of action estimations. While technology exists to automatically create an initial schedule, distribute tasks, and track task completion, the critical monitoring and rescheduling tasks have been, to date, poorly supported [6].

MCM operations involve a unique set of specialized tasks that must be scheduled to minimize the risk to ships from sea mines. What follows is a brief description of the tasks in an MCM operation and their characteristics. The schedule for an MCM operation tasks multiple vehicles to repeatedly *hunt* and/or *sweep* subsections of a specified *threat area* where mines are expected, slowly transiting back and forth in a lawnmower-like search pattern, until the risk of remaining mines is reduced to an acceptably low level. The paths followed by these search vehicles are referred to as *tracks*.

Hunting is a search and destroy activity that encompasses use of specialized sensors to find underwater objects that are *mine-like*, identification of mine-like objects as *mines* or *non-mines*, and neutralization of all discovered mines. The *probability of detection* describes the equipment's sensitivity within that range to the size and reflectivity of mine casings. Because mines may be missed, missions are commonly evaluated according to a *percent clearance* objective. Percent clearance is defined as the probability that a mine at any given position in the search area will be detected.

Sweeping is an activity that uses specialized apparatus to destroy all mines present in a given area either by cutting the chains that connect them to the ocean floor or employing signal generators which mimic the magnetic and acoustic signatures, of ships, to trigger mines that are activated by those signatures.

The operation schedule, which may consist of hundreds of tasks of heterogeneous types, must be repeatedly adjusted over the course of the operation in response to unexpected events which invalidate it. The task of keeping the schedule up to date despite hundreds of interrelated tasks is complex, difficult, and laborious, particularly given the constant time pressure of typical operations. Modifications to schedules are kept to a minimum, in order to reduce expense and opportunities for error; we refer to this characteristic as *minimal operational disruption*. However, modified schedules must also fulfill operational requirements such as percent clearance, time limits, and risk to equipment. These difficult tasks (i.e., monitoring, response, and rescheduling) can be greatly aided by new computational tools.

3 CDMA

One way to reduce the burden on MCM human operators is to help with constant monitoring of disruptions that will impact the mission. Constant monitoring of a vast array of disruption types can be quite difficult. In addition to detecting the disruption, diagnosing the root cause of the problem can be daunting, or easily overlooked. Case-Based Disruption Monitoring and Analyzing (CDMA) within the KRePE architecture handles both disruption monitoring and providing possible root causes.

Case-based reasoning (CBR) is a problem solving paradigm that relies on general cases of a problem domain along with specific domain cases. These cases consist of a mapping between problems and a solution. When a new problem is introduced, generally CBR systems map and provides this new problem to the most similar problem already stored in its case base and provides a solution associated with the known problem. We describe the case representation and the CDMA algorithm in detail in the following subsections.

3.1 CDMA Representation

CDMA uses case-based reasoning for monitoring and analysis of disruptions that will impact an ongoing operation. Based on limited information of the world state, the CDMA algorithm determines if a disruption has occurred. A disruption case in our system are generated manually and consists of five parts: violated expectations, parameters, root cause likelihood, root cause questions and new assumptions.

The case applies when all of the violated expectations are met; and the parameters indicate which variables are applied to a specific problem instance. An example problem representation is shown in Table 1. In this example, there is a disruption where the operator has not heard from the unit within the past 15 minutes while it was out in the field performing a task.

The likelihood and list of root cause questions provide information that can be accessed by an operator through an interactive decision making process. The likelihood provides an apriori probability of how likely a particular root cause is for a given disruption. The root cause tests constitute a set of questions that can help the operator

deduce what is causing the disruption. The parameters defined by the violated expectations populate the variables within the questions, detailing the questions to a specific unit, piece of equipment, etc. If these questions are answered, the likelihoods for the root causes adjust to this information. Using the example from above, Table 1 provides the entire case representation. The new assumptions are a set of suppositions or beliefs as to which root cause explains the disruption. The parameters defined from the violated expectations instantiate the problem information into these new assumptions.

Case Example	
Violated Expectations	current_time(?curTime) ^ unit_last_check_in(?unit,?lastCheck) ^ subtract(?curTime,?lastCheck,?difference) ^ greater_equal(?difference,15.0) ^ unit_assigned_task(?unit,?task) ^ ¬equal(?task,'Unassigned')
Parameters	?curTime, ?unit, ?lastCheck, ?difference, ?task
Likelihood	0.9
Questions	Is ?unit communicating on short-wave?
New Assumptions	unit_capability_failure(?unit, 'Communications', ?lastCheck)

Table 1. Case Representation for CDMA algorithm.

With the use of a standard relational database called the Integrated Rule Inference System (IRIS) [8], CDMA can reuse case(s) in the problem space without having to generate new cases for each set of parameter values. Therefore similarity metrics are not being used. From the example, we do not need to create new cases for each type of equipment or unit, as it can handle all of the parameters. When monitoring detects a disruption, it alerts human operators with a message. The operator then decides the root cause of a given disruption. CDMA adds this confirmed root cause assumptions to the case base providing more information to its case base. These new assumptions trigger schedule repair to occur because the disruption affects the mission.

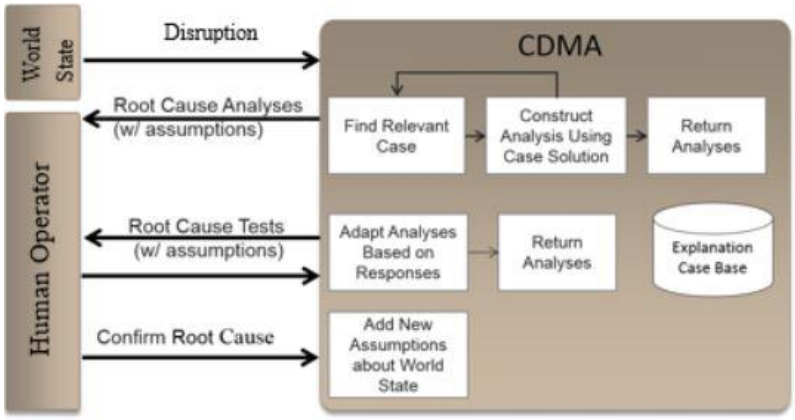


Fig. 1. Workflow for CDMA algorithm.

3.2 CDMA Algorithm

CDMA performs the following steps for disruption monitoring and analysis as shown in Figure 1:

1. ***Find Relevant Case***: To find a possible disruption, CDMA searches through the list of cases to find a relevant case that matches a violated expectation. Each case that matches provides a possible root cause for the disruption.
2. ***Construct Analysis Using Case Solution***: To analyze a disruption, the parameter values indicated by a specific violated expectation are substituted for the parameters specified by an individual case problem.
3. ***Return Analyses***: Each possible disruption is provided on screen for the user to review, detailing the types of root causes for the disruption, along with additional information such as root cause tests and likelihood for each cause.
4. ***Adapt Analyses Based on Responses***: Users can answer these root cause test questions in order for the system to better understand the disruption for future root causes.
5. ***Return Analyses***: The system returns updated likelihoods, sorted with highest likelihood first, along with clearing out infeasible causes.
6. ***Add New Assumptions about World State***: After user selection of the root cause for a disruption, the system creates new assumptions about the world and why the disruption occurred. These new assumptions are added into the case base, providing new information that can be used to generate schedule repair if necessary.

4 CLOSR

To repair schedules that don't meet the criterion of minimal operation disruption, we use the Case-Based Local Schedule Repair (CLOSR) algorithm [10]. This Case base reasoning algorithm in the KRePE architecture creates new assumptions and generates repairs. These repairs strive for "minimal disruption" meaning changes to the schedule should be kept at a minimum while rescheduling to fix a disruption. For example, in MCM operations, repairing a vehicle communication disruption might try to resolve the problem without leaving its search area to minimize transiting time and fuel. Subsequent to case reuse, an adaptation process examines and resolves conflicts created by the schedule repair procedure which is useful for its flexibility. For more detail, please see [10].

5 Evaluation

We hypothesize that the discrepancy monitoring and analysis capabilities of CDMA outperforms ablations that ignore alerts or acts on randomly-selected root causes. To demonstrate this, we ran the CDMA algorithm in an automated manner on a series of simulated MCM operations. For each operation, we measured and compared the performances of three decision makers that: (1) ignores all alerts from CDMA and keeps the original schedule, (2) acknowledges CDMA found disruptions and chooses a random root cause from those suggested therefore rescheduling randomly and (3) acknowledges CDMA found disruptions and chooses the root cause with the highest likelihood. Difference between decision makers indicate the performance improvement that can be achieved by adopting the recommendations made by the CDMA algorithm.

Our study examines an MCM mission with a mine clearing objective. As it is impossible to ensure that 100% of mines are removed in the real world, missions are planned to achieve a high level of percent clearance. This means that there is a high chance that a mine at any given point in the search area would be observed if it existed. The operations conducted in our study are intended to achieve a 95% clearance level; in other words, we would expect 95% of the mines present to be removed. We hypothesize that the decision maker using KRePE's case base will achieve these performance objectives, and that the decision maker that ignores the disruptions will not. This will demonstrate both that monitoring and analyzing disruptions is necessary to achieve an acceptable level of performance under simulated conditions, and that the system is sufficient to achieve that performance.

5.1 Experimental Framework

A simulator for MCM operations, Search and Coverage Simulator (SCSim), another component of KRePE, supports rapid and repeated evaluation and testing of MCM decision support systems and component algorithms. SCSim simulates search missions involving multiple heterogeneous search units, including ships and helicopters, each with different available equipment configurations. Mines and mine-like objects are distributed randomly by SCSim in fields and lines according to pre-set distributions with variable density and object counts. This facilitates evaluation of algorithm performance under varying operating conditions. As a benchmark, automated testing of a two month operation takes less than one minute.

SCSim simulates the assignment of parameterized tasks to units according to a schedule, including transit, sweep, and hunt tasks. Task parameters include, for example, the equipment to use for sweeping, and sensor depth for hunting. To simulate a mission, SCSim automatically generates appropriate tracks for each task and simultaneously changes the position of each vehicle along its assigned tracks. Observations (e.g., contacts) are generated based on vehicles' positions and the sensor equipment in use. Interactions of deployed sweeping equipment is also simulated, and changes the internally represented status of mines. In addition to the scheduled tasks, SCSim is responsible for simulating random events the unexpected difficulties that invalidate an existing schedule (e.g., equipment failure, bad weather, operator errors).

An individual mission test using SCSim is controlled by a scenario description. Scenario descriptions include, at a minimum, the vehicles and equipment available for use, threat areas to be cleared of sea mines, and task areas where vehicles will operate. Other elements of the scenario specify random distributions for mine like objects, mine line placements, and events that may occur. To mimic the real world as closely as possible, SCSim provides only partial observations for the purposes of rescheduling. For example, when a helicopter's communications system fails, its position is no longer reported to the system. As a result, the helicopter appears not to move.

Experiments are driven by a test harness that integrates with SCSim as shown in Figure 2. The test harness generates scenarios defining: the area of operations, available assets, and the ranges of random experimental variables, such as what mine types will be deployed and when events will trigger. The Test Generator applies an appropriate

decision maker that acts as a user of the system. Each decision maker encodes different responses to situations, such as alerts, that arise during the mission simulation. After all simulated missions are complete, the Performance Evaluator tabulates and summarizes these results in a human readable form.

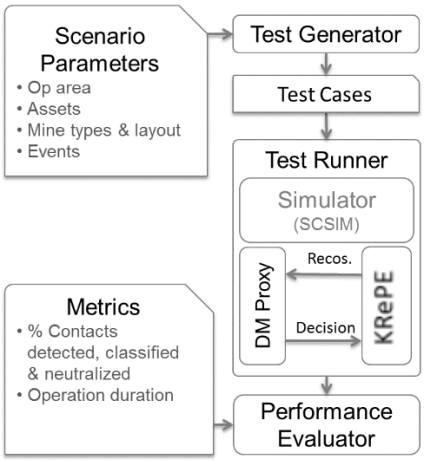


Fig. 2. KRePE simulation driven evaluation

5.2 Experiment Setup

Our experiments used three decision makers and ten randomly generated test scenarios. The first decision maker, "KRePE DM", confirms the correct root cause with the highest disruption likelihoods and selects a new schedule from those generated to activate. The second decision maker, "Random DM", randomly chooses a root cause and selects a new schedule from that root cause. The third decision maker, our baseline, "Ignore DM", ignores KRePE's recommendations, never changing its schedule when prompted. Comparing performance of these three decision makers allows us to measure the efficacy and correctness of schedules generated by case-base disruption monitoring system.

The performance of each decision maker was evaluated in each of ten randomly generated scenarios, generated. (See Table 2). Scenarios differ primarily in the thirty random events that occur and the positions of mines and mine-like objects. Each event was additionally parameterized with a trigger time (chosen randomly over the first six-hundred hours of the mission) and target unit (chosen randomly among the six tasked assets). The times were chosen in this fashion because events that occur when a unit has already performed all its tasks cause no problems, and therefore are uninteresting to our study. Four mine lines, each with a mine count between ten and thirty, at various depths and mine types were placed randomly in each scenario.

The fixed parameters used in all scenarios included the area searched, and seven assets, consisting of four helicopters, two MCM ships, and one support ship that could

assist in tasks if necessary. Each ship and helicopter has available equipment for hunting mines, contact sweeping, detection, and mine neutralization.

5.3 KRePE Metrics

We evaluated KRePE DM, Random DM, and Ignore DM using the following three metrics: (1) *percent contacts detected:* This measures the percentage of mines detected by a unit; (2) *percent mines neutralized:* Percentage all mines are neutralized by a unit and (3) *operation duration:* Total simulation time required to complete the operation.

The first two metrics are calculated based on the true number of mines and mine-like objects generated in the scenario. These summarize the plan's effectiveness in terms of how well the MCM mission goal of searching for and eliminating mines was achieved. Each scenario generated includes a large number of non-mine mine-like objects uniformly spread throughout the threat area, so the percent contacts detected value is an approximation of the percent clearance, or probability that a mine would be detected at any given location. The third metric, operation duration, illustrates a plan's efficiency by measuring the total simulation time required to complete all tasks.

5.4 KRePE Results

Experiments were run on an i7 processor laptop, taking one hour to complete. Figure 3 shows a scatter plot that displays the percentage of existing contacts that were classified correctly and duration of each mission operation measured in simulation hours. The duration of an operation performed by Ignore DM varies little, as the original schedule is never updated, whereas the duration of KRePE DM and Random DM missions can vary greatly. A schedule can be lengthened dramatically when new mine types have been discovered; to ensure safety, many new hunt and/or sweep tasks must be introduced to clear the additional mines. Similarly, if vehicles are damaged beyond repair, the diminished resources can greatly increase mission length. The increased time and repaired schedules allow KRePE DM to outperform Ignore DM by classifying between 95 and 100% of the mine like objects in every mission. Random DM, like KRePE DM, responds to disruptions, but because it does not choose the most likely cause, its task performance is not as high as KRePE DM's. Note that neither Ignore DM nor Random DM represents any real human decision maker; rather these results should be interpreted to show the difficulty of the task and that CDMA's suggestions are benefitting mission performance.

Table 2 shows one-tailed t-test with paired examples. The results include the average and standard deviation for each metric and decision maker. Note: indicate the (small) likelihood that Ignore DM might on average achieve higher values than KRePE DM if many more experiments were undertaken.

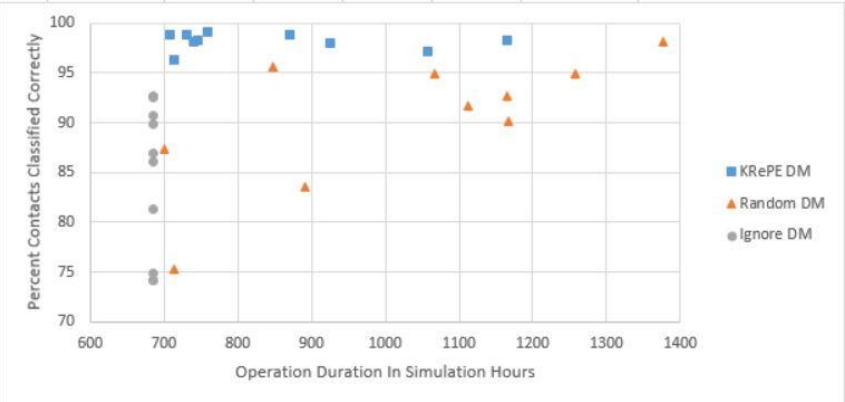


Fig. 3. Scatter Plot of Operation Duration to Percent Contacts Classified Correctly

Table 2. KRePE Results

Metric	KRePE DM	Random DM	Ignore DM
Percent Contacts Detected	98.2 ± **0.8**	90.4 ± **6.5**	84.3 ± **7.2**
Percent Mines Neutralized	93.7 ± **8.1**	88.3 ± **9.6**	81.5 ± **14.4**
Operation Duration	841.6 ± **152.2**	1030.0 ± **218.8**	685.5 ± **0.3**

6 Related Work

Case-based reasoning [1] is a problem solving process based on the adaptation and application of known solutions to new problems. It has been applied to many different domains and problems besides disruption detection.

DISCOVERHISTORY [9] looks for explanations of observations through abductive reasoning, where it maps an observation to a hypothesis that accounts for the observation. DISCOVERHISTORY has been shown to be effective over a large problem space, but is slow with determining disruptions. This is not sufficient for quick detection of immediate issues required by mine countermeasures operations.

A case-based reasoning system, CHEF [7] creates food recipes and explains its own failures. The system tries strategies to see which one can be used to fix the recipe plan. CHEF uses causal rules to explain why its own plan fails. However, the system does not handle constrained resources present in a typical scheduling problem.

The system described in [3] is a CBR system that focuses on wartime equipment maintenance by analyzing feature sets of equipment for maintenance. The system automates the process of deciding the quality of the equipment. CDMA, in contrast, supports a "man-in-the-loop" in order to allow operators to have control over what should be done about disruptions.

7 Conclusion

We presented the CDMA algorithm within the KRePE system that supports monitoring for disruptions and disruption analysis in mine countermeasures operations. Scheduling in this domain is challenging due to the complexities resulting from a large number of tasks that must be allocated over numerous resources. CDMA includes components that assist operation planners by constantly monitoring the environment for changes and providing analysis of discrepancies. Once disruption detection occurred CDMA made it possible for the CLOSR algorithm to reschedule without the need to replan by recommending alternative schedules. We introduced the requirement of minimally disruptive repair as a key operational requirement for automatic schedule repair algorithms in MCM applications.

Our results indicate the efficacy of a case-based strategy; schedule repair was rapid, and created new schedules on demand that ensured the elimination of all mines and increased clearance to a reasonable level. This presents a novel and measurable increase in automated MCM rescheduling capabilities. In the future, we want to apply our system to Unmanned Combat Logistic missions in order to demonstrate effective case-base disruption monitoring with other domains.

8 Bibliographic References

1. Aamodt, A., & Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI communications*, *7*(1), 39-59.
2. Boyd, John, R. (1995). The essence of winning and losing. 28 June 1995.
3. Cai, Jiwei., Jia, Yunxian., Gu, Chuang., and Wu, W. (2011). Research of Wartime Equipment Maintenance Intelligent Decision-making Based on Case-Based Reasoning. In *Procedia Engineering* (Volume 15, 2011, pp. 163-167. CEIS 2011).
4. Chief of Naval Operations, Commandant of the Marine Corps, & Commandant of the Coast Guard. (2007). *A Cooperative Strategy for 21st Century Seapower*.
5. Cummings, Mary, and Collins, Angelo. (2010). Autonomous Aerial Cargo/Utility. In *Concept of Operations, Department of the Navy, ONR, Science & Technology*.
6. Garcia, G. A., & Wettergren, T. A. (2012). Future planning and evaluation for automated adaptive minehunting: A roadmap for mine countermeasures theory modernization. In *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics.
7. Hammond, Kristian J. (1986). CHEF: A Model of Case-Based Planning. In *Proceedings of the Fifth National Conference on Artificial Intelligence*. Philadelphia, Pennsylvania.
8. *IRIS. Program documentation. IRIS Reasoner. Vers. 0.6. N.p., 3 Apr. 2008*. Web. 1 Aug. 2015. <http://www.iris-reasoner.org/pages/user_guide.pdf>.
9. Molineaux, M., Kuter, U. and Klenk, M. (2012). DiscoverHistory: Understanding the past in planning and execution. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multiagent Systems* (pp. 989–996. ACM Press, Valencia).
10. Molineaux, M., Auslander, B., Moore, P. G., & Gupta, K. M. (2015). Minimally disruptive schedule repair for MCM missions. In *SPIE Defense+ Security*. International Society for Optics and Photonics.

A CBR Approach to the Angry Birds Game

Adil Paul and Eyke Hüllermeier

Department of Computer Science
University of Paderborn, Germany
{adil.paul,eyke}@upb.de

Abstract. In this paper, we present a CBR approach for implementing an agent playing the well-known Angry Birds game. We adopt a preference-based procedure for constructing the case base, collecting experience from a random agent that continually explores the problem-solution space and improves the quality of already found solutions. As the retrieve phase involves finding a game scene similar to a given one, we develop a measure to assess the dissimilarity between two game scenes, which is based on solving appropriate linear assignment problems. A comparison of our agent with state-of-the-art computer programs shows promising results.

1 Introduction

Angry Birds is a popular video game, in which the player has to shoot birds from a slingshot at pigs that are protected with objects from different types of materials, including wood, stone, and ice. Some birds have specific capabilities that allow them to explode, split into several birds, pick up speed, etc. The game has different levels, each level coming with its specific representation of pigs and objects hiding them. A level is solved when all the pigs are destroyed, and the goal of a player is to solve all the levels, keeping the number of shot birds as low as possible.

Since the first edition of the Angry Birds AI competition in 2012, different approaches, ranging from qualitative representation and reasoning over simulation of game scenes to classical supervised machine learning algorithms, have been leveraged to build agents playing the game. In this paper, we develop an Angry Birds agent on the basis of the case-based reasoning (CBR) paradigm. To the best of our knowledge, this is the first CBR approach to Angry Birds. One of the main components of our Angry Birds agent is a case base that stores problem-solution pairs, i.e., game scenes and appropriate best shots. We use a preference-based approach to build the case base, which compares different solutions for a given problem and maintains the better one.

The rest of the paper is organized as follows. In the next section, we briefly review some of the existing approaches for agents playing the Angry Birds game. In Section 3, we present our approach, and in Section 4, we analyze its performance experimentally. We conclude our work and outline possible directions for future work in Section 5.

2 Existing Approaches

Most of the work so far has been concerned with the representation of the different types of objects in Angry Birds. Lin et al. [7] classify the objects into dynamic, which are mainly convex polygons, and static ones, which comprise concave polygons, and use bounding convex polygons (BCPs) to represent the former and edge detection and Hough transform to detect the latter. Zhang and Renz [12, 13] also make use of the spatial representation of objects and, moreover, reason about their stability. They build on an extension of the rectangle algebra to assess the stability of blocks of objects, upon which they can decide where to hit a block so as to affect it maximally.

In [11], the authors assign a numerical score to each reachable object, based on its physical properties. The score is supposed to reflect the extent of damage it suffers if being hit, and shoots at objects with low stability but high influence on pigs or shelters of pigs. Ferreira et al. [3] also assign a utility value to the objects based on spatial properties, but because of the lack of certainty in the position of the objects, they incorporate concepts of probability and uncertainty to determine the chance of a bird to hit a given target.

Simulation-based approaches include the work by Polceanu and Buche [9], who build their decision making based on the theory of mental simulation. More precisely, their agent observes the effects of performing multiple simulations of different shots in a given game scene and selects the optimal solution based on these results.

The remaining category of approaches encompasses agents that leverage different machine learning algorithms. In order to learn how to judge shots, Narayan-Chen et al. [8] train a weighted majority and a Naive Bayes algorithm on a data set consisting of good and bad shots in different states of the game. Tziortziotis and Buche [10] use a tree structure to represent the objects in a game scene, and formulate the problem of selecting an object for shooting as a regression problem. They associate with each pair of object material and bird a Bayesian linear regression model, building a competitive ensemble of models, whose parameters are estimated in an online fashion. The decision is then made according to the best prediction of the ensemble model.

3 A Case-based Angry Birds Agent

We employ the CBR approach [1] to build an agent that plays the Angry Birds game. The experience-oriented learning and reasoning paradigm of CBR first of all requires the creation of a case base that stores problem-solution pairs. As the problem space in the domain of Angry Birds is infinite, and no exact characterization of an optimal solution (the best shot) for a problem (a description of a game scene) exists, a way of gathering expressive pairs of problems and approximate solutions (game scenes together with reasonably good shots) is needed. Further, a game scene in Angry Birds comprises objects with different shapes, which should be represented and stored appropriately. Thus, a representation

that reflects the spatial properties of the different objects involved in the game is another concern. Lastly, once the case base is built and appropriately stored, the problem of retrieving cases similar to a given query case needs to be addressed, which in turn necessitates assessing the similarity between two game scenes. In the following, we elaborate on each of these issues.

3.1 Case Base Construction

The core of a CBR system is a case base that stores previously encountered problems and associated solutions. In the context of Angry Birds, a single case should enclose a problem description part, with a representation of a game scene, covering the slingshot and all objects and pigs, and a solution part, containing the best shot one can execute in the given scene. The notion of an optimal solution in a given game scene, i.e., the shot that will lead to the highest change in score, is actually not well-defined. Therefore, we need a procedure to find solutions of at least close-to-optimal quality.

Inspired by the general framework of preference-based CBR [5], we construct a case base by comparing the quality of solutions that have been tried so far. The basic principle of the approach consists of randomly trying different solutions for a problem and maintaining the best one. The advantages of this approach are two-fold. First, because of its self-adaptive nature, it does not rely on any external domain expert to provide solutions for the potentially infinite number of problems. Second, as the problem and solution space are explored more and more, the extent of the case base is enlarged and its quality is improved over time.

In the context of Angry Birds, we concretise the approach as follows. We let arbitrary agents play in different game scenes and record the game scene along with the shot executed by the agent and the change in score. Once we encounter a game scene which is similar to another one already contained in the case base, and where the agent performs better, we replace the solution part of the old case (i.e., the shot) with the new one. The steps of the process of case base construction are outlined in Figure 1 as a flowchart diagram.

3.2 Case Representation

The Angry Birds game involves different types of objects: a sling, hills, pigs, blocks of stone, wood or ice, TNTs and birds with different capabilities expressed in terms of colours, including red, yellow, blue, black, and white. The Angry Birds Basic Game Playing Software [4] provides two possibilities of representing theses objects: the Minimum Bounding Rectangle (MBR) and the real shape representation. While the MBR segmentation of an object consists solely of finding a rectangle with minimal area, which completely covers it, the real shape segmentation represents the objects more precisely using circles, rectangles and polygons, and distinguishes between hollow and solid objects. As such, the latter is more precise but also more costly to compute. In this paper, we confine ourselves to the MBR representation of objects.

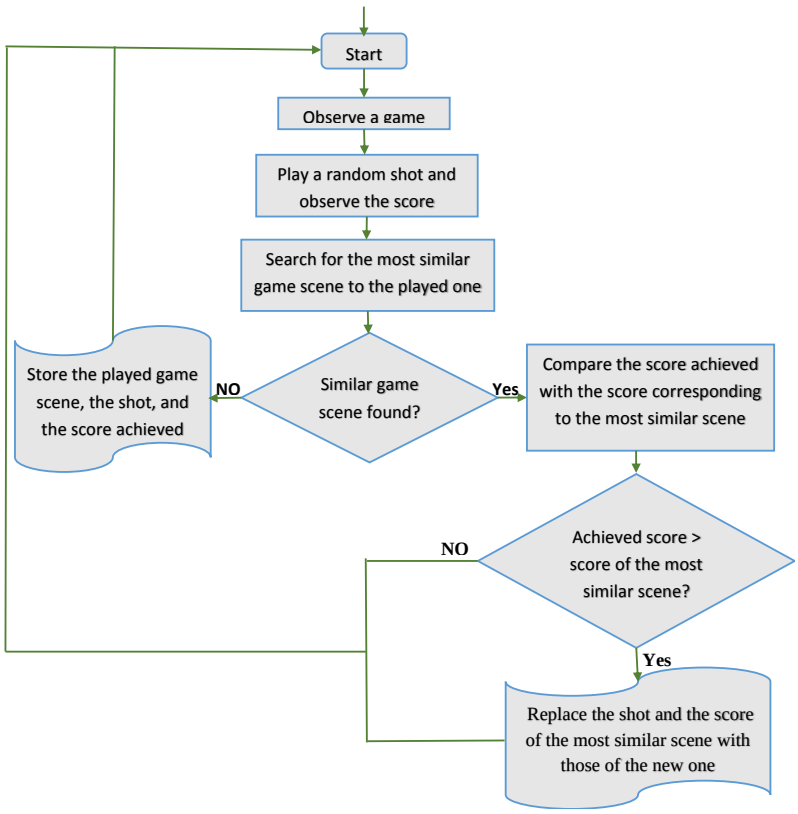


Fig. 1. The steps of the case base construction process.

For describing the rectangles, we adopt the interval-based representation, where a rectangle in the 2-dimensional space $\mathbb{R}^2$ has the following form: $R = [l, u] = [l_1, u_1] \times [l_2, u_2]$, where $l = (l_1, l_2)$ and $u = (u_1, u_2)$ are the coordinates of the lower left and upper right vertex of R, respectively. A complete game scene is represented through the set of the MBRs of all objects, together with their type when an object and colour when a bird.

Besides the game scene, collecting the cases also involves recording shots, which constitute the solution part of a case. In the Angry Birds Basic Game Playing Software, a shot is represented in the form of a 6-dimensional vector $s = (x, y, d_x, d_y, t_{shot}, t_{tap})$, where (x, y) and $(x + d_x, y + d_y)$ are the coordinates of the focus and release point, respectively, t_{shot} specifies the releasing and t_{tap} the tapping time of the bird in milliseconds.

To illustrate how a case is constructed, we consider the situation shown in Figure 2. The start game scene is shown in the picture on the left. The resulting scene after performing the shot with the trajectory indicated by the red line is shown in the picture on the right, where the change in score is seen as well.

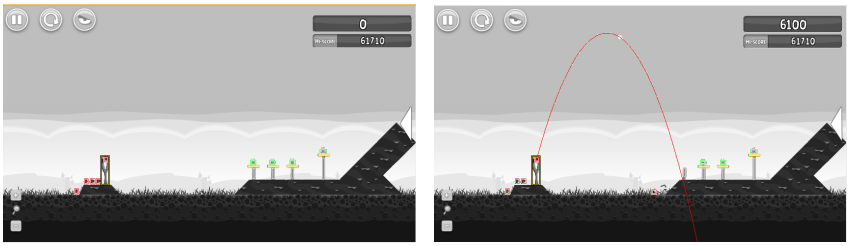


Fig. 2. The game scene before (left) and after (right) performing the shot indicated by the red line in the figure on the right. The MBRs of all objects in both scenes are marked. The change in score after performing the shot is shown on the top right of the figure on the right.

The case extracted from this scenario will contain the original scene, the performed shot and the achieved score, which we represent as follows:

`Sling:` $l_1 = 200,\ u_1 = 305,\ l_2 = 216,\ u_2 = 363.$
`BirdType:` $RedBird.$
`Hills:`
`Hill 1:` $l_1 = 471,\ u_1 = 237,\ l_2 = 839,\ u_2 = 384.$
`Pigs:`
`Pig 1:` $l_1 = 645,\ u_1 = 290,\ l_2 = 659,\ u_2 = 300.$
`Pig 2:` $l_1 = 504,\ u_1 = 314,\ l_2 = 514,\ u_2 = 321.$
`Pig 3:` $l_1 = 543,\ u_1 = 313,\ l_2 = 353,\ u_2 = 323.$
`Pig 4:` $l_1 = 584,\ u_1 = 313,\ l_2 = 595,\ u_2 = 323.$
`TNTs:` -
`Blocks:`
`Block 1:` $l_1 = 651,\ u_1 = 309,\ l_2 = 654,\ u_2 = 352.$
`Block 2:` $l_1 = 509,\ u_1 = 330,\ l_2 = 513,\ u_2 = 351.$
`Block 3:` $l_1 = 548,\ u_1 = 330,\ l_2 = 552,\ u_2 = 351.$
`Block 4:` $l_1 = 588,\ u_1 = 330,\ l_2 = 591,\ u_2 = 350.$
`Block 5:` $l_1 = 643,\ u_1 = 302,\ l_2 = 663,\ u_2 = 304.$
`Block 6:` $l_1 = 500,\ u_1 = 325,\ l_2 = 520,\ u_2 = 327.$
`Block 7:` $l_1 = 540,\ u_1 = 325,\ l_2 = 560,\ u_2 = 327.$
`Block 8:` $l_1 = 579,\ u_1 = 325,\ l_2 = 599,\ u_2 = 327.$
`Shot:` $x = 208,\ \ y = 315,\ d_x = 35,\ d_y = 868,\ t_{shot} = 0,\ t_{tap} = 0.$
`Score:` 6100.

3.3 Case Retrieval

When the agent is playing, it gets a representation of the current game scene, searches the case base for the case with the most similar game scene and adopts its shot. Therefore, an appropriate measure to assess the similarity respectively dissimilarity between two game scenes is a key prerequisite for a successful agent. We compute the overall dissimilarity between two game scenes as the sum of the

dissimilarities between their individual components:

$$\begin{aligned} diss(scene_1, scene_2) = & \, diss(scene_1.Sling, scene_2.Sling) \\ & + diss(scene_1.BirdType, scene_2.BirdType) \\ & + diss(scene_1.Hills, scene_2.Hills \\ & + diss(scene_1.Pigs, scene_2.Pigs) \\ & + diss(scene_1.TNTs, scene_2.TNTs) \\ & + diss(scene_1.Blocks^S, scene_2.Blocks^S) \\ & + diss(scene_1.Blocks^W, scene_2.Blocks^W) \\ & + diss(scene_1.Blocks^I, scene_2.Blocks^I) \ , \end{aligned}$$

where $Blocks^S, Blocks^W$ and $Blocks^I$ denote blocks of stone, wood and ice, respectively.

The dissimilarity of two slings is just the dissimilarity between their MBRs. For the bird type, we compute the dissimilarity as follows:

$$diss(scene_1.BirdType, scene_2.BirdType) = \begin{cases} 0, & \text{if the types are equal,} \\ constant, & \text{otherwise.} \end{cases}$$

Measuring the dissimilarity between two game scenes in each of the remaining components (hills, pigs, TNTs, and blocks) reduces to measuring the dissimilarity between the two sets of rectangles, with potentially different cardinality, corresponding to the MBRs surrounding them. This requires building pairs from the elements of the two sets, between which the dissimilarity is to be computed. The overall dissimilarity between the two sets is then the sum of the dissimilarities between all pairs. We formulate the task of computing the dissimilarity between two sets of rectangles as a (potentially unbalanced) linear assignment problem, where the agents are the elements of one set, tasks are the elements of the other set and the total cost of an assignment is the overall sum of the dissimilarities between all built pairs.

In the following, we proceed with the description of the measure we use for assessing the dissimilarity between two rectangles, prior to detailing our approach to computing the dissimilarity between two game scenes in the above-mentioned components through solving appropriate assignment problems.

Dissimilarity Between Two Rectangles. Different measures exists to assess the dissimilarity between two rectangles in a p-dimensional space. We use the vertex-type distance d_v [2], which is defined for two 2-dimensional rectangles $R_1 = \left[l^{(1)}, u^{(1)}\right] = \left[l_1^{(1)}, u_1^{(1)}\right] \times \left[l_2^{(1)}, u_2^{(1)}\right]$ and $R_2 = \left[l^{(2)}, u^{(2)}\right] = \left[l_1^{(2)}, u_1^{(2)}\right] \times \left[l_2^{(2)}, u_2^{(2)}\right]$, as follows:

$$d_v\left(R_1, R_2\right) = \left(l_1^{(1)} - l_1^{(2)}\right)^2 + \left(u_1^{(1)} - u_1^{(2)}\right)^2 + \left(l_2^{(1)} - l_2^{(2)}\right)^2 + \left(u_2^{(1)} - u_2^{(2)}\right)^2 \ .$$

Dissimilarity Between Two Sets of Rectangles. As stated above, we build on solving an assignment problem to compute the dissimilarity between two sets of rectangles, which represent the MBRs of objects of specific material in two game scenes to be compared.

The linear assignment problem consists of mutually assigning objects of two sets $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_n\}$ in a cost-optimal manner. Formally, assignment costs are defined in terms of a matrix $C = (c_{ij})$, where c_{ij} denotes the cost of assigning a_i to b_j (and vice versa), $i.j \in [N] = \{1, \ldots, N\}$. The goal, then, is to find an assignment that minimizes the total cost

$$\sum_{i \in [N]} \sum_{j \in [N]} c_{ij} x_{ij}$$

with

$$x_{ij} = \begin{cases} 1, & \text{if } a_i \text{ and } b_j \text{ are mutually assigned,} \\ 0, & \text{otherwise.} \end{cases},$$

subject to the following constraints:

$$\sum_{j \in [N]} x_{ij} = 1 \text{ for all } i \in [N],$$

$$\sum_{i \in [N]} x_{ij} = 1 \text{ for all } j \in [N],$$

The Hungarian algorithm [6] is one of the best-known methods for solving the assignment problem. It is mainly based on the observation that adding or subtracting a constant from all the entries of a row or a column of the cost matrix does not change the optimal solution of the underlying assignment problem. Thus, the algorithm proceeds iteratively, subtracting and adding constants in each step to specific rows and columns of the cost matrix, in such a way that more and more zero-cost pairs are built, until an optimal solution can be found. We refer to [6] for a detailed description of the Hungarian algorithm.

In the simplest form of the assignment problem, the number of objects in A and B are equal. For the problem at hand, this assumption does not hold; instead, we are dealing with an unbalanced assignment problem. To handle such problems, one usually introduces dummy rows or columns in the cost matrix, depending on which number exceeds the other. Normally, the introduced entries are filled with zeros, but this does not fit our purpose, because the addition or removal of objects will normally influence the best shot in a scene. We overcome this issue by associating a penalty with objects that remain unassigned. The penalty term for an unassigned rectangle is its distance to the zero-perimeter rectangle located at the origin, i.e., $R = [0, 0] \times [0, 0]$.

4 Experimental Results

We begin our experimental analysis with the construction of the case base, in which we proceed as follows. We run a random agent that chooses the coordinates

of the shot to be executed fully at random, and we restrict ourself to the first 21 levels of the "Poached Eggs" episode of Angry Birds. The agent plays each level several times and the cases from each level are first collected in separate files. The distribution of the number of cases we gathered over the different levels of the game, shown in Table 1, was not uniform. That is, we dedicate more examples to harder levels than to easier ones. At the end, we combine all cases in one file, ending up with a case base of total size of 11,703, which serves as the main case base for our agent.

Table 1. The number of cases we collected in each of the 21 levels of the game.

Level	# cases
1	50
2	50
3	50
4	130
5	50
6	100
7	50

Level	# cases
8	50
9	100
10	647
11	50
12	50
13	182
14	100

Level	# cases
15	50
16	50
17	50
18	400
19	200
20	100
21	100

After the case base was constructed, we first tested the performance of our agent on the above-mentioned levels. To this end, we let the agent play 10 games and report the minimal, maximal, and average score for each level, together with the standard deviation, in Table 2.

To get an idea of how our agent performs in comparison to others, Figure 3 plots the average score of our agent from Table 2 together with the scores of the naive agent, the top-3 agents of the 2013 and 2014 participants of the AI competition, and the average scores of all 30 participants, on all 21 levels, based on the 2014 benchmarks provided on the *aibirds.org* website. This comparison shows that our agent clearly outperforms both the naive and the average agent in both per-level and total scores, and is even competitive to the top-3 agents.

5 Conclusion and Future Work

We made use of CBR to build an Angry Birds playing agent. The results of an experimental study, in which we compared our agent with others, including the top-3 systems of previous AI competitions, are very promising, especially in light of the rather simple implementation of our agent so far. In fact, we are convinced that our agent's performance can be further enhanced through the collection of more cases and the refinement of the different steps of the CBR cycle.

More concretely, this work can be extended along the following directions. First, the real shape instead of the MBR representation can be used to represent the objects involved in the game. Second, a weighted version of the distance measure between game scenes can be learnt. Third, cases from levels of the

Table 2. The minimal, maximal, and average score, and the standard deviation of our agent in 10 games on the first 21 levels of the "Poached Eggs" episode of Angry Birds.

Level	Min. score	Max. score	Mean score	Standard deviation
1	28950	30790	29735	704.955
2	60950	61520	61293	188.388
3	42510	42540	42529	11.005
4	10660	36810	22500	9174.102
5	59680	67760	65301	2302.744
6	18020	35620	32096	6115.800
7	31180	46200	42486	5777.303
8	54110	54120	54111	3.162
9	32130	50020	44525	5874.565
10	32650	59920	46980	9294.536
11	54130	57390	55634	910.668
12	53010	54880	54248	550.713
13	21530	48090	33036	8987.933
14	49250	73760	65553	6858.706
15	37760	48540	46486	3166.492
16	54410	64790	61646	3073.714
17	46290	49900	48492	1224.444
18	39710	60830	49888	7150.137
19	31710	38550	33127	1999.445
20	34030	59140	46527	10113.806
21	59720	96880	70332	11020.633

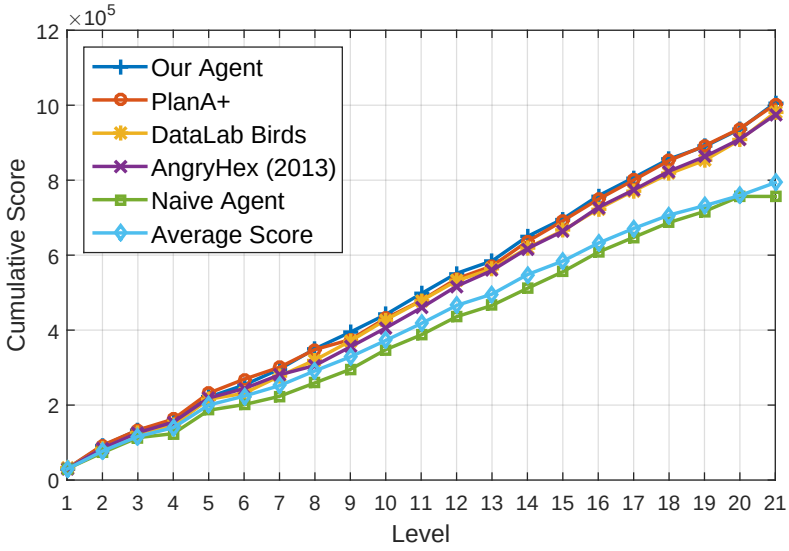


Fig. 3. The cumulative scores of our agent, the top 3 agents of the 2013 and 2014 participants of the AI competition, the naive agent, and the average agent, on the first 21 levels of the "Poached Eggs" episode of Angry Birds.

game other than the ones of the "Poached Eggs" episode can be extracted to increase the size and coverage of the case base. Fourth, since our agent does not realize any adaptation of the retrieved solutions so far, a sophisticated adaptation strategy could be another means to improve performance.

Acknowledgments. This work has been supported by the German Research Foundation (DFG).

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications* 7(1), 3959 (1994)
2. Bock, H.H. Analysis of symbolic data: Exploratory methods for extracting statistical information from complex data. E. Diday (Ed.). *Springer-Verlag New York*, Inc., Secaucus, NJ, USA (2000)
3. Ferreira, L.A., Lopes, G.A.W., Santos, P.E.: Combining qualitative spatial representation utility function and decision making under uncertainty on the Angry Birds domain. In: *IJCAI 2013 Symposium on AI in Angry Birds* (2013)
4. Ge X., Gould, S., Renz, J., Abeyasinghe, S., Keys, J., Wang, A., Zhang, P. Angry Birds basic game playing software, version 1.32. Technical report. Research School of Computer Science, The Australian National University (2014)
5. Hüllermeier, E., Schlegel, P.: Preference-based CBR: First steps toward a methodological framework. In: Ram, A., Wiratunga, N. (eds.) *ICCBR 2011*. LNCS, vol. 6880, pp. 7791. Springer, Heidelberg (2011)
6. Kuhn, H. W. The Hungarian method for the assignment problem. *Naval Research Logistics*, 2: 8397 (1955)
7. Lin, S., Zhang, Q., Zhang, H.: Object representation in Angry Birds game. In: *IJCAI 2013 Symposium on AI in Angry Birds* (2013)
8. Narayan-Chen, A., Xu, L., Shavlik, J. An empirical evaluation of machine learning approaches for Angry Birds. In: *IJCAI 2013 Symposium on AI in Angry Birds* (2013)
9. Polceanu, M., Buche, C.: Towards a theory-of-mind-inspired generic decision-making framework. In: IJCAI 2013 Symposium on AI in Angry Birds (2013)
10. Tziortziotis, N., Papagiannis, G., Blekas, K. A Bayesian ensemble regression framework on the Angry Birds game. In: *ECAI 2014 Symposium on Artificial Intelligence in Angry Birds* (2014)
11. Wałęga, P., Lechowski, T., Zawidzki, M. Qualitative physics in Angry Birds: first results. In: *ECAI 2014 Symposium on Artificial Intelligence in Angry Birds* (2014)
12. Zhang, P., Renz, J. Qualitative spatial representation and reasoning in Angry Birds: first results. In: *IJCAI 2013 Symposium on AI in Angry Birds* (2013)
13. Zhang, P., Renz, J. Qualitative spatial representation and reasoning in Angry Birds: the extended rectangle algebra. In: *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning*, KR14, p. to appear, Vienna, Austria (2014)

Flexible Plan-Subplan Matching for Plan Recognition in Case-Based Agents

Keith A. Frazer[1,3], Swaroop S. Vattam[2], and David W. Aha[3]

[1]College of Computing, Georgia Institute of Technology, Atlanta, GA
[2]NRC Postdoctoral Fellow; Naval Research Laboratory (Code 5514); Washington, DC
[3]Navy Center for Applied Research in Artificial Intelligence;
Naval Research Laboratory (Code 5514); Washington, DC
kfrazer3@gatech.edu | {swaroop.vattam.ctr.in, david.aha}@nrl.navy.mil

Abstract. Plan-subplan matching is an important step in case-based plan recognition. We present RelaxedVF2, an algorithm for plan-subplan matching for plans encoded using the Action Sequence Graph representation. RelaxedVF2 is a subgraph monomorphism algorithm that affords flexibility and error tolerance for plan-subplan matching. We present a study comparing RelaxedVF2 with an alternate degree-sequence matcher that we used in our prior work and found that RelaxedVF2 attains higher plan recognition accuracy on a paradigmatic domain.

Keywords: Plan Recognition, Case-Based Reasoning, Action-Sequence Graph, Relaxed Graph Matching, Error-Tolerant

1. Introduction

An agent on a team must cooperate and coordinate its actions with its teammates, requiring the ability to recognize its teammates' plans. *Plan recognition* refers to the task of observing a teammate's current actions, inferring the plan governing those actions, and predicting that teammate's future actions. A plan recognizer takes as input the observed portion of a plan (subplan) and outputs a (predicted) full plan. A case-based plan recognizer matches its input subplan to a set of plans in its case base and retrieves a most similar plan to the given subplan. We assume that the most similar plan best explains the observed subplan. Plan-subplan matching is therefore a key component of case-based plan recognition (CBPR).

The algorithm used for plan-subplan matching depends on the representation of plans. Typically plans are represented as (a sequence of) propositions in first-order predicate logic. We instead use an *Action Sequence Graph (ASG)* representation (Vattam et al., 2014; 2015), which has some nice properties: (1) it captures the topology of the propositional plans using graphs, (2) better lends itself to vectorization and approximate matching, (3) and makes the matching process more robust to input errors (Vattam et al., 2015). ASG represents a plan as

a labeled directed multigraph. Plan-subplan matching using ASGs reduces to graph-subgraph matching.

We introduce the RelaxedVF2 algorithm for graph-subgraph matching that is tailored to matching plans represented as ASGs. RelaxedVF2 is an extension of the popular VF2 algorithm (Cordella et al., 2004) for subgraph isomorphism. The extensions to VF2 we propose transform it into a subgraph monomorphism matching algorithm, which makes RelaxedVF2 better suited for additional edges that arise between the nodes of states and actions as a plan's observed execution progresses.

In §2 we present related work on CBPR and graph matching techniques. In §3 we present the ASG representation for plans. In §4 we present our plan-subplan matching approach, including the RelaxedVF2 algorithm and the scoring function. In §5 we present an initial empirical study comparing the performance of RelaxedVF2 to an alternative degree-sequence matching algorithm that we used in our prior work (Vattam et al., 2015). Our results show that RelaxedVF2 compares favorably to the alternative approach. We conclude and discuss future research plans in §6.

2. Related Research

Several approaches has been proposed to address the problem of plan recognition (Sukthankar et al., 2014), including consistency-based (e.g., Hong, 2001; Kautz & Allen, 1986; Kumaran, 2007; Lau et al., 2003; Lesh & Etzioni, 1996), and probabilistic approaches (e.g., Bui, 2003; Charniak & Goldman, 1991; 1993; Geib & Goldman, 2009; Goldman et al., 1999; Pynadath & Wellman, 2000). Both types are "model-heavy", requiring accurate models of an actor's possible actions and how they interact to accomplish different goals. Engineering these models is difficult and time consuming. Furthermore, these plan recognizers perform poorly when confronted with novel situations and are brittle when the operating conditions deviate from model parameters.

CBPR is a model-lite, less studied approach to plan recognition. Existing CBPR approaches (e.g., Cox & Kerkez, 2006; Tecuci & Porter, 2009) eschew generalized models for plan libraries that contain plan instances which can be gathered from experience. CBPR algorithms can respond to novel inputs outside the scope of their plan library by using plan adaptation techniques. However, earlier CBPR approaches were not error-tolerant.

In contrast, our work on SET-PR focuses on error-tolerant CBPR (Vattam et al., 2014; 2015). We showed that SET-PR is robust to three kinds of inputs errors (missing, mislabeled, and extraneous actions). One of the factors contributing to its robustness is that SET-PR uses an ASG plan representation and the degree sequence similarity function for plan-subplan matching. Although we previously showed that SET-PR was robust to input errors, there is room for improvement.

VF2 (Cordella et al., 2004) is an exact graph matching algorithm for finding node-induced subgraph isomorphisms. It is one of the few such algorithms applicable to directed multigraphs. Our extension, RelaxedVF2, transforms VF2 from finding node-induced

subgraph isomorphisms to subgraph monomorphisms (see Figure 1 for an illustration on how these differ) and by modifying it to return partial mappings from graph to subgraph when no complete match is available.

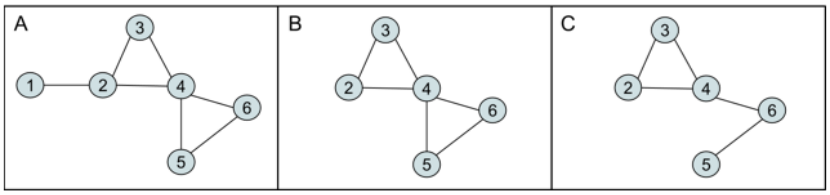


Figure 1: Graph B is a *node-induced isomorphism* of Graph A because it is missing a node (1) but preserves all edges between nodes shared in both Graphs A and B. Graph C is a *monomorphism* of A because it is missing a node (1) and an edge (between 4 and 5). Definitions are provided in Section 4.1.

3. Plan Representation: Action Sequence Graphs

Suppose a plan is modeled as an *action state sequence* $\mathbb{s} = \langle(\boldsymbol{a_0}, \boldsymbol{s_0}), \ldots, (\boldsymbol{a_n}, \boldsymbol{s_n})\rangle$, where each action $\boldsymbol{a_i}$ is a ground operator in the planning domain, and $\boldsymbol{s_i}$ is a ground state obtained by executing $\boldsymbol{a_i}$ in $\boldsymbol{s_{i-1}}$, with the caveat that $\boldsymbol{s_0}$ is an initial state, $\boldsymbol{a_0}$ is null, and $\boldsymbol{s_n}$ is a goal state. An action $\boldsymbol{a}$ in $(\boldsymbol{a}, \mathbf{s}) \in \mathbb{s}$ is a ground literal $\boldsymbol{p} = p(o_1{:}\, t_1, \ldots, o_n{:}\, t_n)$, where $p \in \boldsymbol{P}$ (a finite set of predicate symbols), $o_i \in \boldsymbol{O}$ (a finite set of object types), and t_i is an instance of o_i (e.g., stack(block:A, block:B)). A state $\boldsymbol{s}$ in $(\boldsymbol{a}, \mathbf{s}) \in \mathbb{s}$ is a set of ground literals (e.g., {on(block:A,block:B), on(block:B,substrate:TABLE)}).

An *Action Sequence Graph (ASG)* is a graphical representation of a plan that preserves its topology (including the order of the propositions and their arguments). Vattam et al. (2014; 2015) provide a detailed definition of ASGs and their generation. An ASG is automatically generated by transforming individual propositions in a plan into predicate encoding graphs, and by taking the union of all the individual predicate encoding graphs so as to maintain the total order of the plan. Figure 2 shows an example proposition and its corresponding predicate encoding graph. Figure 3 shows an example full plan and its corresponding ASG. An ASG is a labeled directed multigraph, which constrains the set of graph matching algorithms that can manipulate them.

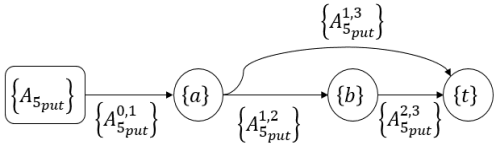


Figure 2: A predicate encoding graph corresponding to $\boldsymbol{p} = put(block{:}\, a, block{:}\, b, table{:}\, t)$

4. Robust Plan-Subplan Matching

As our goal is error-tolerant plan recognition, our approach requires plan-subplan matching that is robust to input errors. Plan-subplan matching requires a measure of similarity. As we encode our plans in the case library and the input subplans as graphs, we utilize maximum common subgraph *monomorphism* as a measure of similarity between them rather than the more conventional maximum common subgraph *isomorphism* measure.

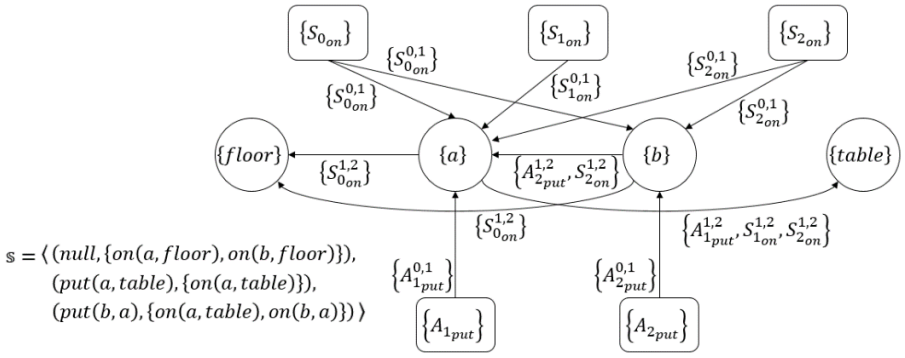


Figure 3: An example of a plan with three action-state sequences and its corresponding ASG

Let G_1, G_2 be graphs composed of sets of vertices and edges V_1, V_2 and E_1, E_2 respectively. G_2 is *isomorphic* to a subgraph of G_1 if there exists a one-to-one mapping between each vertex of V_2 and a vertex in V_1 and the number of edges between nodes in the mapping are maintained. G_2 is instead *monomorphic* if it consists of any subset of the vertices and edges of G_1. Monomorphism must be utilized over isomorphism when matching incomplete subplans to complete plans in the case library because as plans are observed new edges are often added relating existing action and state vertices.

RelaxedVF2 (§4.1), an exact graph matching algorithm, does not return a similarity score. It instead returns a one-to-one mapping of nodes between the subplan and plans in the case library. While the length of the maximum common subgraph is often used to score matches, we instead developed a more nuanced candidate scoring algorithm (§4.2) to increase matching accuracy.

4.1 RelaxedVF2

RelaxedVF2 (Algorithm 1) computes the maximum common subgraph monomorphism between two labeled directed multigraphs. Here we refer to node-induced isomorphism as a subset of the nodes with all corresponding edges between them.

VF2 matches two graphs, G_1 and G_2, using semantic and syntactic feasibility functions to iteratively add compatible nodes of the graphs to an internal mapping, M, which is expressed as a set of pairs (n, m) that represent the mapping of a node $n \in G_1$ with a node $m \in G_2$. Therefore, a mapping M is a *graph* isomorphism if it is a bijective function that preserves

the branching structure of the two graphs, and M is a *subgraph* isomorphism if the mapping is an isomorphism between G_2 and a subgraph of G_1.

The original VF2 algorithm uses five syntactic feasibility rules to check if a pair (n, m) can be included in M. These rules are listed in Table 1.

Table 1: Syntactic feasibility rules for VF2 and RelaxedVF2

VF2	**RelaxedVF2**
$R_{pred}(s,n,m) \Leftrightarrow$ $\left(\forall n' \in M_1(s) \cap Pred(G_1,n) \exists m' \in Pred(G_2,m) \mid (n',m') \in M(s)\right) \wedge \left(\forall n' \in M_2(s) \cap Pred(G_2,n) \exists m' \in Pred(G_1,m) \mid (n',m') \in M(s)\right)$	$R_{pred}(s,n,m) \Leftrightarrow$ $\begin{pmatrix} \forall n' \in M_2(s) \cap Pred(G_2,n) \\ \exists m' \in Pred(G_1,m) \mid (n',m') \in M(s) \end{pmatrix}$
$R_{succ}(s,n,m) \Leftrightarrow$ $\left(\forall n' \in M_1(s) \cap Succ(G_1,n) \exists m' \in Succ(G_2,m) \mid (n',m') \in M(s)\right) \wedge \left(\forall n' \in M_2(s) \cap Succ(G_2,n) \exists m' \in Succ(G_1,m) \mid (n',m') \in M(s)\right)$	$R_{succ}(s,n,m) \Leftrightarrow$ $\begin{pmatrix} \forall n' \in M_2(s) \cap Succ(G_2,n) \\ \exists m' \in Succ(G_1,m) \mid (n',m') \in M(s) \end{pmatrix}$
$R_{in}(s,n,m) \Leftrightarrow$ $(Card(Succ(G\_1,n) \cap T\_1\hat{}in\ (s)) \geq Card(Succ(G\_2,n) \cap T\_2\hat{}in\ (s))) \wedge$ $(Card(Pred(G\_1,n) \cap T\_1\hat{}in\ (s)) \geq Card(Pred(G\_2,n) \cap T\_2\hat{}in\ (s)))$	$R_{in}(s,n,m) \Leftrightarrow$ $(Card(Succ(G\_1,n) \cap T\_1\hat{}in\ (s)) \geq Card(Succ(G\_2,n) \cap T\_2\hat{}in\ (s))) \wedge$ $(Card(Pred(G\_1,n) \cap T\_1\hat{}in\ (s)) \geq Card(Pred(G\_2,n) \cap T\_2\hat{}in\ (s)))$
$R_{out}(s,n,m) \Leftrightarrow$ $(Card(Succ(G\_1,n) \cap T\_1\hat{}out\ (s)) \geq Card(Succ(G\_2,n) \cap T\_2\hat{}out\ (s))) \wedge$ $(Card(Pred(G\_1,n) \cap T\_1\hat{}out\ (s)) \geq Card(Pred(G\_2,n) \cap T\_2\hat{}out\ (s)))$	$R_{out}(s,n,m) \Leftrightarrow$ $(Card(Succ(G\_1,n) \cap T\_1\hat{}out\ (s)) \geq Card(Succ(G\_2,n) \cap T\_2\hat{}out\ (s))) \wedge$ $(Card(Pred(G\_1,n) \cap T\_1\hat{}out\ (s)) \geq Card(Pred(G\_2,n) \cap T\_2\hat{}out\ (s)))$
$R_{out}(s,n,m) \Leftrightarrow$ $(Card(\tilde{N}\_1\ (s) \cap Pred(G\_1,n)) \geq Card(\tilde{N}\_2\ (s) \cap Pred(G\_2,n))) \wedge$ $(Card(\tilde{N}\_1\ (s) \cap Succ(G\_1,n)) \geq Card(\tilde{N}\_2\ (s) \cap Succ(G\_2,n)))$	

The first two rules determine match compatibility based on an equivalent number of incoming and outgoing edges per node. The final three rules look ahead to adjacent nodes to prune the search tree. We adapted VF2 to relax its enforcement of graph/subgraph edge counts while maintaining rules disqualifying additional edges in the subgraph not present in the graph. We removed the fifth rule because strict lookahead rules run counter to our goal of increased error tolerance. We also made modifications to enable returning partial matches, removing the rule that matches must be equal in length to the subgraph.

We optimized RelaxedVF2 for graph recognition, and thus primarily rely on the semantic similarity of node labels to restrict our search space. Our simple semantic feasibility function uses an exact string match of the node labels. Any plan recognizer using this algorithm would need to provide as input its own domain-specific semantic similarity measure. RelaxedVF2

uses a depth-first search through all possible nodes that can be added based on semantic and structural similarity. It then progresses to nodes matching on only semantic similarity before finally adding nodes matching using only structural similarity.

Algorithm 1: RelaxedVF2

PROCEDURE RelaxedVF2(s, G_1, G_2)
  INPUT: An intermediate state s (the initial state s_0 has $M(s_0) = \emptyset$) and two graphs
  OUTPUT: the mappings between G_1 and G_2
  IF $M(s)$ covers all the nodes of G_2 **THEN**
    OUTPUT $M(s)$ //The function M returns the mappings between nodes of G_1, G_2 in state s
  ELSE
    L = [ ] //Sorted list of feasible pairs
    mappingFound = False
    $P(s) \leftarrow$ candidate pairs for inclusion in $M(s)$ //Used candidate pairs function from VF2
    FOREACH $(n, m) \in P(s)$
      IF $F(s, n, m)$ **THEN**
        L $\leftarrow$ L $\cup$ (n, m)
    WHILE NOT mappingFound //Loop until match is found or no more candidates
      $s' \leftarrow M(s) \cup$ L.pop(n, m) //Get the top feasible pair from list
      mappingFound = RelaxedVF2(s', G_1, G_2) //Recursive call
    IF NOT mappingFound //Output partial match if no match found
      OUTPUT $M(s)$
    Restore data structures

4.2 Scoring

The size of the largest common subgraph can be used as a similarity measure (Bergmann, 2002). VF2 is error tolerant and will return matches even if they are of lower quality. Therefore, we designed a metric that is a function of both match size and quality. The match algorithm scores 1 point for every full match based on both semantics and structure (0.7 per semantic match and 0.3 per structural match, based on previous weights used in SET-PR (Vattam et al., 2015)). After retrieving all matches of the subgraph against the case library this score is then used to sort and find the best match.

5. Empirical Study

In this study, we compare plan-subplan matching using two similarity measures on ASGs: (1) RelaxedVF2, and (2) DSQ (degree-sequence matcher) (Vattam et al., 2014; 2015). Our claim is that RelaxedVF2 offers better performance compared to DSQ.

The default plan representation consists of action-state sequences ($\langle(\boldsymbol{a_0}, \boldsymbol{s_0}), \ldots, (\boldsymbol{a_n}, \boldsymbol{s_n})\rangle$). We also evaluated a plan representation consisting of only action sequences ($\langle(\boldsymbol{a_0}), \ldots, (\boldsymbol{a_n})\rangle$) because state information is not always available in all planning domains and it presents a more difficult challenge for DSQ. This yields four conditions: $\text{RelaxedVF2}_{\text{ActionStates}}$, $\text{DSQ}_{\text{ActionStates}}$, $\text{RelaxedVF2}_{\text{ActionsOnly}}$, and $\text{DSQ}_{\text{ActionsOnly}}$.

We empirically test whether, for error tolerant CBPR, using RelaxedVF2 outperforms DSQ for both the ActionStates and ActionsOnly conditions. We use plan recognition accuracy as our performance metric, where accuracy is defined as the ratio of queries that resulted in correct plan retrieval to the total number of queries.

5.1 Empirical method

We conducted our experiments in the Blocks World domain, which is simple and allows us to quickly generate a plan library L with desired characteristics. We used SHOP2 (Nau et al., 2003) to generate L's plans as follows. We generated 20 random initial states and paired each with 5 randomly generated goal states to obtain 100 planning problems. Each was given as input to SHOP2 to obtain a plan. We fixed plan length to 20 by discarding any whose length was not 20 and generating a new one (with a different goal state) in its place. This distribution was chosen because it is challenging for plan recognition.

We evaluated plan recognition accuracy using a leave-one-in strategy (Aha & Breslow, 1997). For each compared condition:

1. We randomly selected a plan π in L (π is *not* deleted from L).
2. We introduced a fixed percentage of error into π consisting of a uniform distribution of missing, mislabeled, and extraneous actions and random distortions of states associated with those actions. The error levels that we tested were {0%,10%,20%,30%,40%,50%}.
3. The error-π plan was then used to incrementally query L to retrieve a plan. For example, if error-π had 20 steps, the evaluator performed 11 queries at the following plan lengths: 0% (initial state only, no actions are observed), 10% (first two actions and states are observed), and so on until 100% (full plan is observed).
4. Each query derived from error-π was used to retrieve the top matching plan π^{sol}. If π was equal to π^{sol}, it was considered a success and a failure otherwise.
5. We repeated steps 1-4 for all 100 plans in L in each of 20 trials.

This yields 1100 queries per error percent level per trial, yielding 132,000 queries (1100 queries × 6 error levels × 20 trials). We computed average accuracy over 20 trials.

5.2 Results and discussion

We computed mean accuracy for each percentError (in [0.0,0.5] with increments of 0.1) and each percentAction (in [0.0,1.0] with increments of 0.1) for RelaxedVF2 and DSQ. The results are shown in Figures 4 and 5 for ActionStates and ActionsOnly, respectively. Our results show that for all error levels and percent actions RelaxedVF2's mean accuracy was higher than DSQ's. In ActionStates, RelaxedVF2 achieves 50% accuracy by 20% actions at all error levels, but DSQ only achieves 50% accuracy at 100% actions at only 0% and 10% error. In ActionsOnly, RelaxedVF2 achieves 50% accuracy by 40% actions at all error levels, but DSQ only approaches 50% accuracy at 100% actions with 0% error.

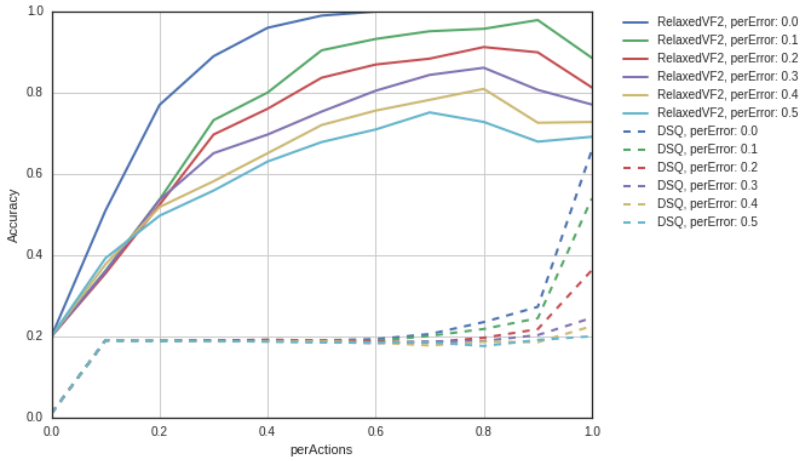


Figure 4: Mean plan recognition accuracy for the ActionStates conditions

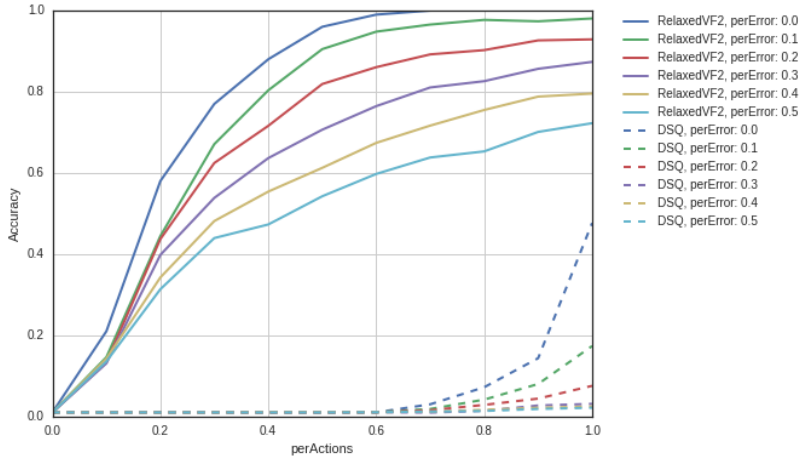


Figure 5: Mean plan recognition accuracy for the ActionsOnly conditions

We conducted a one-way ANOVA test to compare the effects of percent Actions (0%-100%), percent Error (0%-50%) and the matching algorithms (RelaxedVF2, DSQ) on accuracy. There was a significant effect on accuracy at $p < 0.05$ with respect to the matching algorithms ($F(1,17)=18687550.204$, $p=0.0$). This analysis shows that RelaxedVF2 significantly outperformed DSQ, which lends support to our claim.

DSQ performs considerably worse without state information because the ASGs become much smaller. The degree sequences across the partitions of the smaller graphs will yield similar values, preventing DSQ from disambiguating the different plans.

Surprisingly, accuracy decreased around 80% actions in RelaxedVF2 in the ActionStates condition (Figure 4). As the error level increases this dip occurs earlier. We hypothesize that the more densely connected graphs resulting from additional action-state information causes these graphs to more closely resemble each other, thus reducing recognition accuracy. We plan to investigate this in future work.

Given that RelaxedVF2 is an exact graph matching algorithm and DSQ is an approximate algorithm, DSQ should have a significantly shorter runtime. In this study, RelaxedVF2 had a mean runtime (in seconds) of 0.121 and 0.045 in the ActionStates and ActionsOnly conditions, respectively. DSQ mean runtime was 0.020 and 0.019 in these conditions. We subjected the mean runtimes to a *t*-test and found the differences in the runtimes to be significant at $p < 0.05$ for both conditions.

6. Summary

CBPR under imperfect observability requires error tolerant plan-subplan matching, which requires flexible representation and matching algorithms. In earlier work we introduced the ASG representation for plan recognition and degree-sequence plan matching (Vattam et al., 2014; 2015). Although this matching algorithm worked reasonably well, there remained room for improvement. Here we presented RelaxedVF2, an alternative plan-subplan matching algorithm. It is a subgraph monomorphism algorithm, and thus affords flexibility and error tolerance in matching compared to VF2. In our empirical study we found support for our claim that, for error-tolerant CBPR, RelaxedVF2 can outperform the degree-sequence matcher, at least for the paradigmatic domain we studied.

In future work, we will investigate whether the same result occurs when using datasets from additional domains to address the single dataset limitation of our current study. We also plan to integrate RelaxedVF2 into our plan recognition architecture to complement the existing methods. We also plan to do a comparative study with other state-of-the-art plan recognizers.

Acknowledgements

Thanks to OSD ASD (R&E) for sponsoring this research. Swaroop Vattam conducted this research while an NRC post-doctoral research associate at NRL. Keith Frazer conducted this research while an NREIP intern at NRL. The views and opinions in this paper are those of the authors and should not be interpreted as representing the official views of NRL or OSD.

References

Aha, D. W., & Breslow, L. A. (1997). Refining conversational case libraries. *Proceedings of the Second International Conference on CBR* (pp. 267-278). Providence, RI: Springer-Verlag.

Bergmann, R. (2002). *Experience management: Foundations, development methodology, and Internet-based applications*. Berlin, Germany: Springer-Verlag.

Bui, H. (2003). A general model for online probabilistic plan recognition. *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence* (pp. 1309–1315). Acapulco, Mexico: Morgan Kaufmann.

Charniak, E., & Goldman, R. (1991). A probabilistic model of plan recognition. P*roceedings of the Ninth National Conference on Artificial Intelligence* (pp. 160–165). Anaheim, CA: AAAI Press.

Charniak, E., & Goldman, R. (1993). A Bayesian model of plan recognition. *Artificial Intelligence*, *64*, 53–79.

Cordella, L., P., Foggia, P., Sansone, C., & Vento, M. (2004). A (sub)graph isomorphism algorithm for matching large graphs. *Transactions on Pattern Analysis and Machine Intelligence*, *26*(10), 1367-1372.

Cox, M.T., & Kerkez, B. (2006). Case-based plan recognition with novel input. *Control and Intelligent Systems*, *34*(2), 96–104

Geib, C.W., & Goldman, R.P. (2009). A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, *173*(11), 1101–1132.

Goldman, R.P., Geib, C.W., & Miller, C.A. (1999). A new model of plan recognition. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence* (pp. 245–254). Bled, Slovenia: Morgan Kaufmann.

Hong, J. (2001). Goal recognition through goal graph analysis. *Journal of Artificial Intelligence Research*, *15*, 1–30.

Kautz, H., & Allen, J. (1986). Generalized plan recognition. *Proceedings of the Fifth National Conference on Artificial Intelligence* (pp. 32–37). Philadelphia, PA: Morgan Kaufmann.

Kumaran, V. (2007). *Plan recognition as candidate space search*. Master's thesis, Department of Computer Science, North Carolina State University, Raleigh, NC.

Lau, T., Wolfman, S.A., Domingos, P., & Weld, D.S. (2003). Programming by demonstration using version space algebra. *Machine Learning*, *53*(1-2), 111–156.

Lesh, N., & Etzioni, O. (1996). Scaling up goal recognition. *Proceedings of the Fifth International Conference on Knowledge Representation and Reasoning* (pp. 178–189). Cambridge, MA: Morgan Kaufmann.

Nau, D., Au, T.-C., Ilghami, O, Kuter, U, Murdock, J.W., Wu, D., & Yaman, F. (2003). SHOP2: An HTN planning system. *Journal of Artificial Intelligence Research*, *20*, 379-404.

Pynadath, D.V., & Wellman, M.P. (2000). Probabilistic state-dependent grammars for plan recognition. P*roceedings of the Conference on Uncertainty in Artificial Intelligence* (pp. 507–514). San Francisco, CA: Morgan Kaufmann.

Sukthankar, G., Goldman, R., Geib, C., Pynadath, D., & Bui, H. (Eds.) (2014). *Plan, Activity, and Intent Recognition*. Cambridge, MA: Elsevier.

Tecuci, D., & Porter, B.W. (2009). Memory based goal schema recognition. In *Proceedings of the 22nd International Florida AI Research Society Conference*. Sanibel Island, FL: AAAI Press.

Vattam, S.S., Aha, D.W., & Floyd, M. (2014). Case-based plan recognition using action sequence graphs. *Proceedings of the Twenty-Second International Conference on Case-Based Reasoning* (pp. 495-510). Cork, Ireland: Springer.

Vattam, S., Aha, D.W., & Floyd, M. (2015). Error tolerant plan recognition: An empirical investigation. In *Proceedings of the Twenty-Eighth Florida Artificial Intelligence Research Society Conference*. Hollywood, FL: AAAI Press.

Experience and Creativity

Workshop at the
Twenty-Third International Conference on
Case-Based Reasoning
(ICCBR 2015)

Frankfurt, Germany
September 2015

Raquel Hervás and Enric Plaza (Editors)

Chairs

Raquel Hervás	Universidad Complutense of Madrid, Spain
Enric Plaza	IIIA-CSIC, Spain

Program Committee

Kathleen Agres	Queen Mary, University of London, UK
Amílcar Cardoso	Univesidade de Coimbra, Portugal
Belén Díaz-Agudo	Universidad Complutense of Madrid, Spain
Pablo Gervás	Universidad Complutense of Madrid, Spain
Maarten Grachten	Austrian Research Institute for Artificial Intelligence, Austria
Maria Teresa Llano	Goldsmiths, University of London, UK
Ramon López de Mántaras	IIIA-CSIC, Spain
Santiago Ontañón	Drexel University, USA
Rafael Pérez y Pérez	Universidad Autónoma Metropolitana, Mexico
Senja Pollak	Josef Stefan Institute, Slovenia
Hannu Toivonen	University of Helsinki, Finland
Martin Znidarsic	Josef Stefan Institute, Slovenia

Additional Reviewers

Pascal Reuss	University of Hildesheim, Germany
Viktor Ayzenshtadt	University of Hildesheim, Germany

Preface

We are pleased to present the proceedings of the ICCBR-15 Workshop on Experience and Creativity, which was held as part of the 23rd International Conference on Case-Based Reasoning in Frankfurt am Main, Germany, September 2015 with the cooperation of PROSECCO, the European Network for Promoting the Scientific Exploration of Computational Creativity.

The relationship between past examples in a domain and computational creativity in that domain is an interesting and essential topic that has not been explicitly addressed. The goal of the workshop is to address common areas of interest in Case-Based Reasoning (CBR) and Computational Creativity (CC) by addressing research issues that are related to both communities. In order to do so, our goal is to explore and analyze how the new and innovative (creativity) is related to, depends upon, and needs to break away to from the old and know (experience). The main focus of the workshop will be on exploring the relationship between past examples in a domain and computational creativity in that domain.

The workshop provided a forum for discussion of these new research directions, provoked by the presentation of five long papers and four position papers, which are collected in these proceedings. Geraint Wiggins gave an invited talk for the workshop on the relation of experience and creativity. Znidarsic, Tomašič and Papa presented a case-based approach to automated generation of slogans, including a methodology for evaluation and ranking of the final results, which indicate the ability of the approach to create valuable slogan prototypes. Gervás, Hervás and León presented a case-based reasoning solution that builds a plot line to match a given query, expressed in terms of a sequence of abstraction of plot elements of a story, by retrieving and adapting templates for narrative schemas from a case-base. Hervś, Sánchez-Ruiz, Gervás and León compared the judgement on similarity between stories explained by a human judge with a similarity metric for stories based on plan refinements, taking into account that is difficult to compute between complex artifacts such as stories. Gonçalves, Martins, Cruz and Cardoso proposed an evolutionary high performance algorithm that extracts two semantic sub-graphs from a knowledge base to be used as building blocks in computational blending processes. Pollak, Martins, Cardoso and Urbancic investigated which principles people use when they name new things as results of blending, with the aim of uncovering patterns with high creative potential and to use them for automated generation of names for new creations or phenomena. Valitutti discussed ideas for characterizing the re-use of procedural knowledge, performed by a case-based generative system, as creative. The implied idea is to characterize as creative the search path that allows the system to discover new basins of attraction. Agres stated that there is a clear connection to be made between psychological findings regarding learning and memory and the areas of case-based reasoning and computational creativity, aiming to encourage researchers in these areas to consider psychological perspectives while developing the technical and theoretical aspects of their computational systems. Cardoso and Martins proposed that conceptual blending, an important

mechanism in computational creativity, can play a role within the case-based reasoning paradigm as an alternative adaptation mechanism that may provide suitable solutions in computational creativity setups. Cunha, Martins, Cardoso and Machado focused on computational generation of visual symbols to represent concepts, aiming to develop a system that uses background knowledge about the world to find connections among concepts with the goal of generating symbols for a given concept.

Finally, we would like to thank everyone who contributed to the success of this workshop, especially the authors, the program committee members, PROSECCO, the organizers of the ICCBR 2015 conference and Joseph Kendall-Morwick, ICCBR Workshop Chair.

September 2015 Raquel Hervás
Frankfurt Enric Plaza

Automated blend naming based on human creativity examples

Senja Pollak[1], Pedro Martins[2], Amílcar Cardoso[2], and Tanja Urbančič[13]

[1] Jožef Stefan Institute, Ljubljana, Slovenia
[2] CISUC, DEI, University of Coimbra, Coimbra, Portugal
[3] University of Nova Gorica, Nova Gorica, Slovenia
senja.pollak@ijs.si, pjmm@dei.uc.pt, amilcar@dei.uc.pt
tanja.urbancic@ung.si

Abstract. In this paper we investigate which principles people use when they name new things as results of blending. The aim is to uncover patterns with high creative potential and to use them for automated generation of names for new creations or phenomena. We collected examples with a web survey in which participants were asked to evaluate pictures of animals with blended anatomies from two different animals, and to provide their own names for blended creatures on the pictures. The blended animals served as a trigger of human creativity manifested through imaginative, humorous, surprising names collected in the survey. We studied how the features from the pictures reflected in the names, what are different complexity levels of lexical blend formation and how far in other realms subjects "travelled" to search for associations and metaphors used in the names. We used the findings to guide automated generation of names for the blends.

Keywords: Computational creativity, human creativity examples, conceptual blending, lexical blend generation, creative naming, bisociation.

1 Introduction

Creativity is in the core of many human activities and has been studied for decades [9][2]. As a phenomenon challenging for being replicated with machines, it became also a topic of artificial intelligence research [21]. While creativity is an intriguing research question by itself, it is also a driving force of development and as such, it has an immense value for applications in countless areas, including scientific discovery, engineering inventions and design. One of the cognitive principles underlying such discoveries and inventions is conceptual blending [5] in which two mental spaces integrate into a new one, called blend. Conceptual blending has also been implemented and tested in computer systems to produce novel concepts [17]. However, there are still many open questions related to the choice of input mental spaces and the ways of projections that lead to blends, perceived as creative and inspiring. In our work we aim at providing guidance

for choosing input spaces and projections based on concrete findings about human creativity with elements of blending. More precisely, by investigating the patterns that we can find in the cases of human creations, we guide the blending process to the extent allowing for automated generation of blends.

Conceptual blending and case-based reasoning [10] can meet in a very fruitful way in areas such as design and architecture [4][6].In such domains, blends are not only a source of surprise, artistic satisfaction or inspiration, but have also their own functionality, bringing into the process additional constraints and priorities. Contexts and goals can also be used in computational approaches to conceptual blending and can beneficially affect the issues of efficiency [13]. Authors in [1] exploit a principle of creative transfer from one domain to another in the realm of design. Their IDEAL system abstracted patterns from design cases in one domain and applied them to design problems in another domain. connecting distant, self-consistent and usually not connected frames of reference has been recognised and used as an effective principle in the act of creation. Such connections of habitually incompatible domains through common patterns or bridging concepts are also referred to as bisociations [9].

In this paper, we address the issue of case-based reasoning and conceptual blending in the context of lexical creativity. While this might appear quite far from the discussion on design in the previous paragraph, the connection becomes evident based on an observation by Veale and Butnariu [20]: "Words are everyday things, as central to our daily lives as the clothes we wear, the tools we use and the vehicles we drive. As man-made objects, words and phrases are subject to many of the same design principles as the consumer artefacts that compete for our attention in the market-place.". The authors also draw attention to two basic principles of artefact design, as identified in [15], namely visibility and mapping. In the case of a well-designed product, the design should suggest a mental visualisation of a conceptually correct model of the product, and the mapping between appearance and function should be clear. Their Zeitgeist system [20] can automatically recognise neologisms produced as lexical blends, together with their semantic meaning. This is done based on seven different "design patterns" recognised in constructing neologisms as lexical blends. Types of lexical blends and how new lexical blends are formed is described and illustrated with many examples in [12]. An important issue of recognising and quantifying creativity in different combinations of words is studied in [11].

In our work we investigate how humans approach the task of naming new things, and how based on human examples, a computer system could exhibit similar (and, why not, better) performance. We consider this principle of using past examples for revealing patterns to be used for new cases as a manifestation of case-based reasoning. The concrete task was to name creatures – animals with blended anatomies from two different animals. This was done in a web-based survey, designed primarily for a study of human perception of visual blends [14]. In this paper we continue using the material of the same study, but we examine it from a completely different angle, i.e. from the lexical creativity side by investigating creative naming of blends. Many offers for supporting naming of

snorse
(snake, horse)

chimporse
(chimpanzee, horse)

durse
(duck, horse)

guineabear
(guinea pig, bear)

hammerheadhorse
(hammerhead shark, horse)

pengwhale
(penguin, whale)

proboscis parrot
(proboscis monkey, bird)

chamelephant
(elephant,chameleon)

duckphant
(elephant, duck)

guinea lion
(guinea pig, lion)

horbit
(horse, rabbit)

hammerhead gull
(hammerhead shark, gull)

horduck
(horse, duck)

spider pig
(spider, guinea pig)

shark retriever
(shark, labrador retriever)

Fig. 1. Hybrid animals dataset used in the online questionnaire (available at http://animals.janez.me). Each sub-caption contains a name of the blend proposed by survey participants, as well as the input spaces. All blends were created by Arne Olav, with the exception of *shark retriever* and *camalephant*, whose authorship is unknown. For a better visualisation, some images were slightly cropped.

client's enterprises, products, etc. can be found on the web and show the application potential of creative naming. The task has already been approached with the goal of (semi-)automatic name generation and the results presented in [16] and [18] demonstrate a very big potential. While our work shares some of the ideas with above-mentioned related approaches, it differs from them by using visually triggered human examples as examples used for automatic lexical blend generation, and by using a novel categorisation of creativity level that guides construction of blends based on bisociation as one of the key principles inherent in many human creative processes.

After presenting the survey in which the names were collected in Section 2, we analyse different patterns and mechanisms used by people when coining names in form of lexical blends in Section 3. These patterns are used in Section 4 for automatically generating blends of different levels. In Section 5 we discuss the potential of our prototype and present further research perspectives.

2 Survey: Visual blends and their lexical counterparts

In [14], we introduced a survey consisting of an on-line questionnaire related to the quality of visual blends. Around 100 participants assessed 15 hybrid animals which were the result of blending anatomies from two different animals (Figure 1).The participants were asked to to rate criteria related to the coherence of blends as well as creativity.

Clearly in our questionnaire on animal blends the main focus was on visual blends. However, with the aim of getting more insight into potential connections, participants were also asked to provide a name (in English, Portuguese, Slovene, French or Spanish) to each of the hybrid creatures. By asking people to name the creatures we wanted to investigate the following questions: Would participants give names for all, for none, or for some of the creatures? How creative are they when naming the animals, how does the visual blended structure reflect in the lexical blend? Where the names provided by subjects mostly lexical blends or not? Do lexical blends use animal's "prototype" characteristics, or more sophisticated associations for which some background knowledge is needed (like titles of books, movies, history, etc.)? Does complexity of visual blends reflect in the names? The names given to the visual blends are the focus of our study.

In our survey we collected 1130 names for 15 animals. The general trend was that people gave more names at the beginning of the study and the trend of the number of given names was descending. However, some pictures triggered more generated names than expected by their position (e.g., *guinea lion* and *spider pig*). The guinea lion is also the blend for which the unpacking (recognising the input spaces) was the most difficult [14] and the one for which the highest number of very creative, bisociative lexical blends were formulated.

3 Formation and complexity of lexical blends

Our previous investigations of relationship between conceptual blending and bisociation have drawn our attention to different levels of blend complexity. To deal with this issue in a more systematic way, we suggest the following categorisation regarding the input words used to form the name:

L1 each of the words appearing in the lexical blend is a commonly used word for one input animal (no mapping);
L2 both input words represent input animal in a rather common way, but are blended into one word by *portmanteau* principle, i.e. by using the prefix of one word and the suffix of the other word (possibly with some intersection);
L3 one word represents one input animal with a commonly used word for this animal, the other word represents a *visible characteristic* (part, colour etc.) of the other animal (variant L3*: both words use such characteristics);
L4 one word represents one input animal with a commonly used word for this animal, the other word represents a characteristic of the other animal for which *background knowledge* about this animal (habitat, way of moving, typical behaviour) is needed (variant L4*: both words use such characteristics);
L5 one word represents one input animal with a commonly used word for this animal, the other animal is represented with a more sophisticated association – *bisociation* – for which a creative discourse into another realm (e.g. from animals to literature) is needed (variant L5*: both words represented with such associations).

We illustrate the categories by the names actually given in the survey to the blended animal *guinea bear*:

L1 *mouse-bear* (input1: *mouse*, input 2: *bear*);
L2 *rabbear* (input1: *rabbit*, input 2: *bear*);
L3 *small-headed bear* (input1: *mouse* → *small head*, input 2: *bear*);
L4 *scared bear* (input1: *mouse* → *scared*, input 2: *bear*);
L5 *mickey the bear* (input1: *mouse* → *Mickey the mouse*, input 2: *bear*).

As seen from this example, while the bear was easily recognised as one of the constituting animals, there were different interpretations about the second animal, "contributing" the head to the blended creature. In fact, the variety in the whole dataset was even bigger as names given by different subjects suggested the second animal being a mouse, rabbit, hamster, guinea pig, rat, squirrel, wombat or opossum. The set of input words as used by the subjects is even bigger since it includes also diminutives, slang versions, etc.

The levels increasing indicate the increasing complexity (but not necessarily the quality) of the blends, but note that they do not build on just one criterion in a linear way and there might also be a combination of principles described at different levels present in one name. We illustrate this with a name *teddybbit*, generated as a portamanteau (L2), but using an association between bear and teddybear from the toys realm (L5).

However, we plan to improve this by introducing a creativity score in which not just the level of mappings used will be taken into account, but also the fact whether they were used for one or for both input animals, and how creative the combination was (e.g., by taking into account phonetic features or by recognizing references extrinsic to the two input animal spaces and their bisociations).

Note that not all of the names provided by the subjects in the survey were lexical blends. Here we do not analyse such names in more detail, but to study the potential for triggering creativity, they are important as well. Some examples collected in our survey for the *guinea bear* are *creepy*, *giant*, or *fluffy*.

4 Patterns from examples for automated name generation

We investigated how the above-mentioned categories of human-generated names could be used for automatic blend generation. Different categories represent different mechanisms. Names of Level 1 are very basic and easy to be automatically generated, their creativity level is low and the name can hardy be called a blend. On the other hand, higher levels (3-5) rely on human experience, background knowledge, associations and bisociations. To generate the names of levels 3 and 4, we use a large web corpus (the enTenTen corpus [7]) and the sketch grammar relations available in Sketch Engine [8]. For the last category (level 5), we used other resources of human knowledge (Wikipedia, imdb lists). For each category, we reveal the patterns in human given names and explain how they can be used in automatic generation. Our generated examples are all done by modifying only one animal name.

L1: In names given by humans, we found two different patterns at level 1. In each case, the two animals are used, the possible variations being either hyphen to indicate the combined meaning "animal1-animal2" (e.g. *dog-shark*) or creating a single word containing full names of both animals "animal1animal2" (e.g. *spiderrat*). The pattern with a premodifier of adjective can be recognised in the given name *mišasti medved*, where the first word is an adjective formed from the noun *miš* (Eng. *mouse*) and the second one is the noun medved (Eng. *bear*). Some word formations are language specific, e.g. in Slovene bare "noun-noun" word formation is not very productive.

To illustrate the automatic name generation, we took the animal names from each input space and concatenated them. Using these simple patterns resulted in names very much resembling those generated by humans, e.g. *duck-horse* or *duckhorse*. More examples are in the L1 row in Table 1.

L2: Level two uses the *portmanteau* principle. In all the languages used in the survey this mechanism was used very frequently. For recognising these names from the list, we focused on words composed of the beginning of one animal word and ending of the other. Examples of basic portmanteau names given by the subjects are the names given in Figure 1. We automatically recognized L2 blends by combining pairs of animals and some simple heuristics.

In automatic generation, the starting point was to combine half of the each of the two input animal names. If the input word consists of two words, frequently

Table 1. Automatically generated names - examples for four fictional animals.

Input level	elephant & chameleon	snake & horse	horse & chimpanzee	duck & horse
L1	elephant-chameleon	snake-horse	horse-chimpanzee,	duck-horse
	elephantchameleon	snakehorse	horsechimpanzee	duckhorse
L2	elepheleon	snarse	horanzee	ducrse
L3	tusk chameleon	venom horse	hoof chimpanzee	beak horse
	trunk chameleon	fang horse	mane chimpanzee	arse horse
	graveyard chameleon	tail horse	bridle chimpanzee	back horse
	tail chameleon	poison horse	rump chimpanzee	feather horse
	ear chameleon	belly horse	withers chimpanzee	
L4	Asian chameleon	venomous horse	Trojan chimpanzee	Anaheim horse
	giraffe chameleon	poisonous horse	wild chimpanzee	lame horse
	captive chameleon	garter horse	Arabian chimpanzee	Peking horse
L5	Dumbo chameleon	Ser Hiss horse	Alfonso chimpanzee	Donald horse
			Daffy horse	Howard the horse

in the analysed examples one word is kept to from the blended name (which is not a proper portmanteau anymore). This pattern was used for generating examples like *guinea lion*, *hammerhead eagle*, *hammerhead goose*.

One could make different combinations based on different proportions of the input words or by using phonetic rules (vowels, consonants, rhymes), exact vs. inexact matching, pronunciation information, word's Greek or Latin origins, etc. as in many advanced existing systems proposing portmanteau name generation [19] [18] [3].

L3: In the next category of lexical blends, humans use visible characteristics of one animal and associate them to the other animal. The properties of the animal that gives the "head" to the new visual blend can be lexically expressed as prepositional phrase modifying the head noun, i.e. the name of the animal providing the body (*horse with snake head, elephant of the orange beak*), by adjective modifier (e.g. *nosy robin, duckbilled pachyderm, trunkheaded chameleon)*or in noun-noun constructions (e.g. *nosebird*). In some cases both animals are described by their characteristic visible parts (e.g. *tail-trunk*). Combinations with portmanteau structure is also possible (e.g. *grivasti kabod* [Eng. *mane horswan*]).

For automated blend generation of L3 we currently use only noun-noun constructions. We rely on the Skecth Engine tool by using word sketches constructed with Sketch grammar. Word sketches are automatic corpus-derived summaries of a word's grammatical and collocational behaviour [8]. From the word sketch of animal "contributing" the head to the visual blend (e.g. *elephant* in Figure 1), we use all the collocators (above selected frequency and salience threshold)

from the grammatical category *possessed*. This lists contains nouns that in the enTenTen corpus follow the search word and 's, e.g. for *elephant's* the list contains *tusk, trunk, ...* resulting from collocation *elephant's tusk* in the corpus. We construct then noun-noun blends, by adding the animal name of the animal providing the body (e.g. *chameleon*). As shown in Table 1, examples using this structure often correspond to parts of the body, (*tusk chameleon, trunk chameleon, tail chameleon, ear chameleon*), while *graveyard chameleon* does not represent the part of the body. Obviously, some of the compounds are irrelevant, e.g. *tail chameleon* – since chameleons have a tail themselves so this description does not contribute anything in terms of blending. Neither does the corpus provide the information if the "possessed" part is located on the animal's head and even less if it corresponds to the depicted picture (e.g. tusks are not depicted on the picture of elephant and chameleon from Fig. 1, even if they are prototypical part of elephant's head). More specific filters and knowledge bases will be used in future to narrow the choice to better candidates.

L4: Level 4 names are more diverse and require more background knowledge. As mentioned in Section 3, the observed categories are habitat, locomotion (*plavajoči konj* [Eng. *swimming horse*], typical behaviour (e.g.*elequack* using animal sounds) or usage (*saddleducks*. Again, also both animals can be represented by their properties, such as in the blended name *galloping quack*. For automated name generation at this level, we used again the word sketches, but we took the information from category *modifiers* (typical adjectival or noun collocators modifying the animal providing the head to the blended creature). E.g. adjectives *venomous* and *poisonous* are typical collocators of word *snake* and are used for forming blended names *venemous horse* and *poisonous horse*. Often breed names are used in modifier position; by selecting only lower case modifiers we can keep more general properties. For Level 4 , more background knowledge is needed. E.g., from automatically constructed names *Trojan chimpanzee, wild chimpanzee* or *Arabian chimpanzee*, the first one is referring to specific cultural reference Trojan horse and can be interpreted at level 5. Same goes for the *lame horse*, which is formed from the idiom *lame duck* (i.e.*an elected official who is approaching the end of his tenure, and esp. an official whose successor has already been elected* (Wikipedia)).

L5: In analysis of human lexical blends we manually classified in Level 5 the bisociative blends using characters from cartoons (*Spider Gonzalez*), children songs (*Slonček Raconček* refering to a Slovene song *Slonček Jaconček*), where slonček means small elephant and *raconcek* comes from *duck – raca*), movies (*My little mallard*), politicians (*Sharkozy*), legends (*Jezerski Pegasus* [Eng. *river Pegasus*]) and often combinations of several of them, e.g. character from movie and from comic strips *Jumbo Zvitorepec* (where Jumbo refers to the animal, while *Zvitorepec* is a character from Slovene comic strip by Miki Muster, but literally means curled tail which refers also to the visual representation of this animal (cf. picture elephant, chameleon in Fig. 1).

For automatically generating highly creative lexical blends inspired by the examples given by participants, we based the bisociative blend generation on

characters from the movies representing the input animal. We created a short list from Wikis, IMDB and Wikipedia pages about animal characters in movies where the last section covers cultural representations. In the name generation process, we first checked if character's name contains the name of the animal and if so we substituted this name with the name of the other input animal (e.g. horse substituting the duck in *Donald horse*). On the other side, if the animal does not appear explicitly we added the name of the second animal to the existing character name (*Dumbo chameleon*). In future, we will expand generation of names at this level by exploring other realms besides movies and books.

5 Discussion

We investigated the principles of creating lexical blends based on visual blends (blended animals). We revealed different mechanisms used in name formation and introduced a new categorisation of blend complexity (L1-concatenation blends; L2-portmanteaux; L3-blending based on visible characteristics; L4- blending using background knowledge and L5-bisociative blends). After the analysis of examples generated names by humans, we made a prototype system for automated generation of blends of different levels using word combinations, grammatical and collocational information and background knowledge resources. The most frequent mechanism used by humans was the portmanteau principle. But a portmanteau can vary from very basic ones to the bisociative ones, since blend strategies can easily be combined. For instance, the blend *shagull* can be interpreted as a simple portmanteau blend (shark+gull) or as bisociative blend refering to Chagall. This example shows that the bisociation can be used on the production level (e.g. creative blend but the reader cannot decompose it), on the interpretation level (e.g. even if there was no such intention when generating a name, the bisociation can be present at the reader's side) or both.

We like some names generated as lexical blends more than the others – what counts? Even if names are generated using similar principles, some of them are much more creative, achieving higher degree of creative duality, compressing multiple levels of meaning and perspective into a simple name [20]). It is the combination of simplicity and bisociation (in our case the switch from animal wor(l)d to cultural realm) that seems to be the most impressive. To verify this claim and to get a more thorough evaluation of automatically generated names, we plan to collect human subjects feedback as well as compare human-generated and automatically generated names. We will also further elaborate the automatic recognition of blend complexity and on the other side the blend generation part (e.g. including phonological criteria, rhymes, more background knowledge, etc.). Next, we will investigate the role of emotions: while some names were neutral, many had very strong emotional content (cf. negative emotions in *disgusoarse, horrabit* or the name given to the *hammerhead gull*, for which instead of naming it a user wrote "deserves death by fire, not a name") or positive emotions in *le trop joli*, name used for guinea lion. Another spectre of research is to investigate the generality of our blend categorisation by applying it to other domains.

This work has been supported by the projects ConCreTe (grant nb. 611733), WHIM (611560) and Prosecco (600653) funded by the European Commission, FP 7, the ICT theme, and the FET program. We thank also A. Fredriksen, the author of the pictures.

References

1. Bhatta, S.R., Goel, A.: Learning generic mechanisms for innovative strategies in adaptive design. The Journal of the Learning Sciences 6, 367–396 (1997)
2. Boden, M.: The Creative Mind: Myths and Mechanisms. Basic Books (1991)
3. Deri, A., Knight, K.: How to make a frenemy: Multitape FSTs for portmanteau generation. In: Proc. of the NAACL (2015)
4. Domeshek, E., Kolodner, J.: A case-based design aid for architecture. In: Gero, J., Sudweeks, F. (eds.) Artificial Intelligence in Design '92, pp. 497–516 (1992)
5. Fauconnier, G., Turner, M.: Conceptual integration networks. Cognitive Science 22(2), 133–187 (1998)
6. Goel, A.K., Craw, S.: Design, innovation and case-based reasoning. The Knowledge Engineering Review 20(3), 271–276 (2005)
7. Jakubíček, M., Kilgarriff, A., Kovář, V., Rychlỳ, P., Suchomel, V., et al.: The tenten corpus family. In: 7th Int. Corpus Linguistics Conf. pp. 125–127 (2013)
8. Kilgarriff, A., Rychly, P., Smrz, P., Tugwell, D.: The sketch engine. In: Proc. EURALEX 2004. pp. 105–116 (2004)
9. Koestler, A.: The Act of Creation. New York:Macmillan (1964)
10. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann Publishers Inc. (1993)
11. Kuznetsova, P., Chen, J., Choi, Y.: Understanding and quantifying creativity in lexical composition. In: Proc. Conf. Empirical Methods in Natural Language (EMNLP). pp. 1246–1258 (2013)
12. Lehrer, A.: Blendalicious. Lexical creativity, texts and contexts pp. 115–133 (2007)
13. Li, B., Zook, A., Davis, N., Riedl, M.O.: Goal-driven conceptual blending: A computational approach for creativity. In: Proc. of the 3rd Int. Conf. on Computational Creativity. pp. 9–16 (2012)
14. Martins, P., Urbančič, T., Pollak, S., Lavrač, N., Cardoso, A.: The good, the bad, and the AHA! blends. In: Proc. of the 6th Int. Conf. on Computational Creativity (2015)
15. Norman, D.A.: The design of everyday things. Basic books, NY (1988)
16. Özbal, G., Strapparava, C.: A computational approach to the automation of creative naming. In: Proc. of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1. pp. 703–711 (2012)
17. Pereira, F., Cardoso, A.: The horse-bird creature generation experiment. The Interdisciplinary Journal of Artificial Intelligence and the Simulation of Behaviour(AISBJ) 1(3), 257–280 (2003)
18. Smith, M.R., Hintze, R.S., Ventura, D.: Nehovah: A neologism creator nomen ipsum. In: Proc. of the 5th Int. Conf. on Computational Creativity. pp. 173–181 (2014)
19. Veale, T.: Tracking the lexical zeitgeist with WordNet and Wikipedia. In: Proc. European Conf. on Artificial Intelligence (ECAI). pp. 56–60 (2006)
20. Veale, T., Butnariu, C.: Harvesting and understanding on-line neologisms. Cognitive Perspectives on Word Formation pp. 399–420 (2010)
21. Wiggins, G.A.: Towards a more precise characterization of creativity in AI. In: Proc. of Workshop Program of ICCBR – Creative Systems: Approaches to Creativity in AI and Cognitive Science (2001)

Generating Plots for a Given Query Using a Case-Base of Narrative Schemas

Pablo Gervás[1], Raquel Hervás[2], and Carlos León[2]

[1] Instituto de Tecnología del Conocimiento, Universidad Complutense de Madrid
Ciudad Universitaria, 28040 Madrid, Spain
pgervas@ucm.es
[2] Facultad de Informática, Universidad Complutense de Madrid
Ciudad Universitaria, 28040 Madrid, Spain
raquelhb@fdi.ucm.es,cleon@ucm.es
WWW home page: http://nil.fdi.ucm.es/

Abstract. Computational generation of literary artifacts very often resorts to template-like schemas that can be instantiated into complex structures. With this view in mind, the present paper presents a case-based reasoning solution that builds a plot line to match a given query, expressed in terms of a sequence of abstraction of plot-bearing elements of a story, by retrieving and adapting templates for narrative schemas from a case-base. The abstractions of plot-bearing elements of a story are defined in terms of Propp's character functions. The case-base of narrative schemas is built based on a review of a number of existing attempts to provide an elementary set of patterns for basic plots. A selection of these patterns, reformulated in terms of Propp's character functions, is used as case-base. The paper explores a solution for automatic generation of stories based on this formulation of the narrative schemas.

Keywords: computational creativity, narrative, narrative schemas, transformational case adaptation, compositional case adaptation

1 Introduction

Humans that write stories reuse material from stories they know. This may include characters, settings, scenes, or lines of dialogue. Of these, the most important is the reuse of story structure. In order to capture computationally this type of reuse of experience, an abstract representation of story structure is needed. The present paper describes a case-based solution for story generation that relies on Vladimir Propp's Morphology of the Folk Tale [13]. A case-base of narrative schemas described using this representation [5] is used to provide plot lines to match a query, and the plot lines are then fleshed out into full stories by instantiating the abstract plot line with specific story actions [4].

2 Previous Work

To support the approach followed in this paper, four areas of previous work need to be considered: case-based approaches to story generation, Propp's formalism

for analysing stories, the Propper system for story generation, and existing approaches to case adaptation.

2.1 Case-Based Approaches to Story Generation

Roger Schank stated that the way in which memory works is not only based on processes that manipulate mental data, but instead as continuous recalling and adapting process of previous stories that define our world [18, 17]. Turner's MINSTREL exemplified this approach by generating short stories about King Arthur and his Knights of the Round Table [19]. MINSTREL handled episodic memories in two different ways: either by instantiating a matching schema in the story from a basic query, or by performing a basic adaptation on the query, querying the episodic memory with it and returning an adaptation and modification of the query. Knowledge intensive case-based reasoning approaches [3, 10, 11] use Semantic Web technologies for knowledge representation and simple combinatorial algorithms for generating the structure of new plots by reusing fragments of structure of previous stories, inspired in the morphology of Russian folk-tales studied by Vladimir Propp [13]. Relying on more shallow representations, [14] and [16] introduce a story planning algorithm inspired by case-based reasoning that incorporates *vignettes* – pre-existing short narrative segments – into the story being generated. Other approaches to story generation based on case bases of previous schemas include efforts towards incorporating analogy-based reasoning to knowledge acquisition [9, 15]. These systems are usually focused on the retrieval, adaptation and evaluation of old schemas to new domains. In general, all these approaches rely on inter-domain analogies and generate new instances of old narrative schemas. Reuse of previous stories is also applied in [12], where case-like structures known as Story Contexts are mined from a set of previous stories and used to inform the selection of the next action to add to a story in an incremental generation process.

2.2 Proppian Morphology of a Story

At the start of the 20th century, Vladimir Propp [13] identified a set of regularities in a subset of the corpus of Russian folk tales collected by Afanasiev [1]. These regularities he formulated in terms of *character functions*, understood as acts of the character, defined from the point of view of their significance for the course of the action. According to Propp, for the given set of tales, the number of such functions was limited, the sequence of functions was always identical, and all these fairy tales could be considered instances of a single structure. The set of character functions identified by Propp includes a number of elements that account for a journey (`departure`, `return`), a number of elements that detail the involvement of the villain and the struggle between hero and villain (`villainy`, `struggle`, `victory`, `pursuit`, `rescue_from_pursuit`), a

number of elements that describe the acquisition of a magical agent by the hero (`test_by_donor`, `hero_reaction`, `acquisition_magical_agent`).[3]

2.3 The Propper System

The Propper system developed by Gervás [4] constitutes a computational implementation of a story generator initially based on Propp's description of how his morphology might be used to generate stories.

It relies on the following specific representations for the concepts involved:

- a *character function*, a label for a particular type of acts involving certain named roles for the characters in the story, defined from the point of view of their significance for the course of the action
- a sequence of character functions chosen as backbone for a given story
- possible instantiations of a character function in terms of specific *story actions*, involving a number of *predicates* describing events with the use of *variables* that represent the set of characters involved in the action

Based on these representations the Propper system defines a procedure that first chooses a sequence of character functions to act as abstract narrative structure to drive the process, and then progressively selects instantiations of these character functions in terms of story actions to produce a conceptual representation – in terms of an ordered sequence of predicates – of a valid story. This conceptual representation is a *fabula*, a sequence of states that contain a chain of story actions – which are instances of those character functions. A story action involves a set of preconditions – predicates that must be present in the context for continuity to exist –, and a set of postconditions – predicates that will be used to extend the context if the action is added to it. Each story action is linked to its context of occurrence by having its preconditions satisfied by the preceding state.

2.4 Case Adaptation

Probably one of the most difficult processes in the CBR cycle is the reuse or adaptation stage. After retrieving the most similar case (or cases) from the case base, the solution from the retrieved case must be used to create a new solution for the problem at hand.

Wilke and Bergman [20] established a classification of CBR adaptation into three different methods: null adaptation, transformational adaptation and generative adaptation. The simplest kind of adaptation is *null adaptation*, where the solution of the retrieved case is used without any modification. As simple as this adaptation method is, it can obtain very good results for simple problems. *Transformational adaptation* consists on the transformation of the solution of

[3] For reasons of space, only a number of character functions relevant to the examples given in the paper are described. Readers can check the referenced sources for more detail.

the retrieved case into the solution required for the query. In order to do that, the retrieved solution may be reorganized and modified by deleting or adding new elements. Finally, *generative adaptation* consists on generating the new solution from scratch, but reusing the process used to obtain the solution from the retrieved case.

These three adaptation methods are formalized by considering that only one case is retrieved and adapted. However, some problems may be better solved by reusing information from more than one case. This is what Wilke and Bergman called *compositional adaptation*, where the new solution is obtained by adapting the solutions of multiple cases. This multiple case adaptation can be done using transformational or generative methods, but the main idea is that the solution for the case at hand can be better obtained by taking into account more than one case from the case base.

There are many examples of compositional adaption in recent CBR works. Arshadi and Badie [2] apply this adaptation in a tutoring library system. In this kind of application it is probable that many cases can be similar to the user request at the same time, so it is important to take all of them into account when generating the solution for a given query. Hervás and Gervás [6] also use multiple cases for text generation based on templates. When the information that must appear in a sentence is not covered by the template of the retrieved case, a new retrieval process is triggered in order to find more cases which templates can cover the information in the query. Ontañón and Plaza [8] present the concept of amalgam as a formal operation over terms in a generalization space. Although amalgams are not proposed as an adaptation method by themselves, the notion of amalgam is related to merging operations that can be used in compositional adaptation to combine two or more cases. Müller and Bergmann [7] use a compositional adaptation approach for cooking recipes represented as cooking workflows. During the adaptation stage, missing parts of retrieved cooking workflows are covered using information from other cases.

3 Case-Based Construction of Plot Lines for Stories

The present paper describes a case-based approach to the construction of plot lines for stories – described as sequences of character functions – which can then be fleshed out into stories.

3.1 Case-Based Construction of Plot Lines

The system operates from a query provided by the user. This query is expressed as a sequence of character functions that the user would like to see included in the desired plot line.

The system compares the given query with the set of plot lines represented in its case base.

The Case-Base The case base of schemas used for this paper is built from the narrative schemas reviewed in [5]. These correspond to a set of sequences of character functions – in Propp's sense of plot relevant abstractions of the activity of characters – that correspond to a number of theoretical characterizations of possible plots for stories, also referred as *narrative schemas*. The case-based reasoning approach will therefore operate over sequences of character functions, and it will return a sequence of character functions that best matches the given query.

Merging Plot Lines When dealing with plot lines in terms of sequences of character functions it is often necessary to merge two plot lines to obtain a third plot line. Because plot lines are sequentially ordered, and specific elements in the plot may have dependencies with other elements, the relative order in which they appear in the sequence is very relevant. For the purposes of the present paper, this is done as follows:

- the query is traversed sequentially
- each character function in the query is checked against the next character function in the case
- if they match the character function is added to a *matching* subsequence
- if they do not, the character function from the query is added to a *wanted* subsequence, and the next character function from the query is checked against the character function in the case
- if the end is reached for the query the rest of the case is added as an *added* subsequence
- if the end is reached for the case the rest of the query is added as a *wanted* subsequence

The merge is constructed by concatenating into a single sequence the subsequences of character functions that are generated during the merge in this fashion. This has the advantage of interleaving the character functions from the original query with the contributions from the various cases involved while always respecting the order in which these character functions appeared in the query.

Similarity We consider a similarity function for plot lines based on identifying the relative mutual coverage between query and case. The set of subsequences of the query that appear as subsequences of the case in the corresponding order is referred to as the *match*. The *remainder* is the set of subsequences of the query that are not covered by the case. The *addition* is the set of subsequences of the case that did not appear in the query.

The similarity employed in the current version of the system is calculated as an average between the percentage of the query covered by the case – the ratio between the size of the match and the size of the query – and the percentage of the case that is involved in the match – the ratio between the size of the match and the size of the case. This is intended to capture the suitability of the

case both in terms of maximum coverage of the query and in terms of minimum addition of character functions beyond the query.

To compute these values the query is merged with the case as described above. The match is then reckoned to be the set of *matching* subsequences. The remainder is then reckoned to be the set of *wanted* subsequences. The addition is then reckoned to be the set of *added* subsequences.

Retrieval and Adaptation If there is a case whose plot line matches the query, that case is returned as solution.

Otherwise, the cases are ranked based on their similarity with the query. The set of character functions that appears in the overall set of subsequences resulting from this process constitutes a possible solution to the problem posed by the query, as it would constitute a combination of the query and the case.

If the retrieved case does not cover all the character functions in the query, further retrieval processes will be required. This corresponds to solving the given query with a complex story that combines more than one plot line. To achieve this, an additional retrieval process is set in motion using the remainder of the first retrieval process as a query to the second one.

For each additional case retrieved, the resulting solution is merged with the result of prior stages using the same procedure as for merging a query and a case. These ensures that relative order of appearance of related character functions within each narrative substructure that has been reused is respected in the final solution.

The retrieval and adaptation process can be iterated until the remainder of the query is empty. The merge obtained at this point is the final solution. This sequence of character functions is the solution found by the system as plot outline for a story to match the given query.

An Example of Plot Line Construction For a query `villainy departure villain_punished return`, the most similar case retrieved is:[4]

`villainy` *`hero_pursued rescue_from_pursuit struggle victory`* `villain_punished`

The merge of the query and case, with the different subsequences marked[5] is:

`villainy DEPARTURE` *`hero_pursued rescue_from_pursuit struggle victory`* `villain_punished RETURN`

Within the resulting merge, the elements not matched by the retrieved case (the remainder: `departure return`) appear in the same relative position with respect to the other elements of the query as they did in the original sequence of the query.

[4] Elements in the case that match the query are shown in plain text, and elements that do not are shown in italic.

[5] `Matched` elements are shown in plain text, `WANTED` elements in small caps, and *`added`* elements in italic.

To cover this remainder, a second case-based reasoning process is set in motion, with the remainder as a query. For this second process, the query would then be DEPARTURE RETURN. The most similar case retrieved is:[6]

departure *difficult_task task_resolved* hero_pursued rescue_from_pursuit struggle victory *test_by_donor hero_reaction acquisition_magical_agent* return

The merge of this additional case with the result of the prior CBR process, with the different subsequences marked as above is

villainy departure *difficult_task task_resolved* hero_pursued *rescue_from_pursuit struggle victory* villain_punished *test_by_donor hero_reaction acquisition_magical_agent* return

This implies that the remainder is now empty.

3.2 Fleshing out the Plot Line for the Story

Because character functions are abstractions of plot relevant activities by the characters, the draft plot line obtained as a result of the retrieval and adaptation stage needs to be fleshed out before it can be considered a story.

This involves instantiating the character functions with specific story actions. This can be done following the original procedure for the Propper system [4] for obtaining a fabula from the sequence of character functions corresponding to the resulting plot line. This relies on definitions of the story actions defined in terms of predicates that define an action, with identifiers for the characters as arguments. The definitions of these story actions also contain predicates that define preconditions and effects of the action in question. The instantiation procedure relies on unification of each new story action with the previous context to guarantee continuity and coherence in terms of which characters perform which actions.

Table 1 presents an example of story corresponding to the plot line obtained as a result of the case-based reasoning procedure described in section 3.1.

It is worth noting that although the character functions being instantiated arise from two different original plot lines as provided by the cases, the fleshing out procedure instantiates them with story actions that link up to conform a single coherent story about a hero (character id147) and a villain (character id755). An initial villainy (state 0) forces the hero to set out (state 1), he faces a difficult task (states 2-3), he undergoes several conflicts with the villain (states 4-5 and 6-7). The end of this particular story involves a meeting with a donor that provides a magical agent (states 9-11) and an eventual return of the hero (state 12).

4 Discussion

The approach followed for case adaptation in the described procedure is transformational and compositional. Both the transformation of the retrieved cases to

[6] The use of italics shows the match of the retrieved case with the sequence resulting from the earlier CBR process.

State	Event description
0	character id755 kidnap id755 id756 victim id756 character id756 misbehaved id755
1	seeker id147 character id147 sets_out id147
2	sets id181 id147 id183 character id181 id183 difficult_task id183 involves id183 manufacture
3	character id181 solve id147 id183 before dead_line
4	runs_away id147 pursues id755 id147 hides_in id147 tree tries_to_destroy id755 tree
5	jumps_to_another tree escapes id147

State	Event description
6	weight_contest id147 id755 confrontation id147 id755
7	makes id147 protective_gesture banishes id755
8	pardoned id755
9	shows id388 id389 donor id388 character id388 magical_agent id389 offers_exchange id388 id389 id147 test id388 id147
10	agrees_to_exchange id147 uses id147 id389 id388 deceives id147 id388 positive_result id147
11	helper id53 character id53 meets id755 id53 offers_services id53 id755
12	returns id147

Table 1. An example story corresponding to the plot line shown earlier.

better match the query and the composition of more than one case are covered by the described procedure for merging two sequences of character functions while respecting the relative order of appearance of their elements.

The procedure followed for story construction operates at a higher level of abstraction than [19, 14, 16], and with greater flexibility than [3, 10, 11] – who also use character functions – due to its highly compositional approach to case recombination.

The case-based reasoning procedure described relies on cases to provide a complete backbone for a plot line, reusing the structure of a given plot completely, with no option for leaving out certain parts of it. The procedure for successive retrievals, together with a merging approach that respects the relative order in which character functions occur in the query and interleaves the additions without repetition, allow for more than one such plot backbone to be combined into more complex stories. However, this approach will only succeed as long as there exists some case in the case base with a reasonably similar sequence of character functions. Beyond this, it might be necessary to consider alternative approaches that allow reuse of fragments of cases, to be recombined into longer sequences.

The choice of case base employed here is built from schemas that are intended as complete plots. Alternative formulations of the case base are possible, built from smaller units of plot, such as scenes. These might be represented as subsequences of character functions that occur frequently in different plot lines. A solution along these lines might define the case base in terms of smaller units that would be abstracted during the construction of the case base. This procedure is similar to the one employed in [12], where cases are retrieved to generate the actions of the story one by one (one case per action). An alternative procedure would be to operate over a case base of complete plots but define

a different retrieval algorithm that allows a certain fragmentation of these plots during retrieval.

Two important aspects to consider in creative plot generators are coherence and novelty. By virtue of its process of reusing large segments of existing plots, the described procedure is likely to generate coherent plots, though how coherence is affected by the merging procedure should be addressed in further work. In that sense, the process of instantiation with story actions employed by the Propper system presents an advantage in that it checks the satisfaction of preconditions of each action in its context during construction. With respect to novelty, processes that reuse existing solutions are exposed to the risk of reproducing aspects of prior material. To address this risk, future work should consider establishing limits on the extent of reuse considered. These could take the form of avoiding cases that are perfect matches for a given query, and preferring solutions obtained by combination of more than one case.

5 Conclusions

The case-based reasoning solution described in this paper operates at a sufficiently high level of abstraction to allow the construction of valid plot lines by combination of cases that represent narrative schemas which are merged into a plot line that matches a given query, and which can then be instantiated into a specific coherent story.

Acknowledgements

This paper has been partially supported by the project WHIM 611560 funded by the European Commission, Framework Program 7, the ICT theme, and the Future Emerging Technologies FET program.

References

1. Alexander Nikolayevich Afanasyev. *Narodnye russkie skazki A. N. Afanaseva [Folk Russian tales of A. N. Afanasev]*, volume 1-3. Moscow, 1855.
2. Niloofar Arshadi and Kambiz Badie. A compositional approach to solution adaptation in case-based reasoning and its application to tutoring library. In *Proceedings of the 8th German Workshop on Case-Based Reasoning, Lammerbuckel*, 2000.
3. P. Gervás, B. Díaz-Agudo, F. Peinado, and R. Hervás. Story Plot Generation Based on CBR. *Knowledge-Based Systems. Special Issue: AI-2004*, 18:235–242, 2005.
4. Pablo Gervás. Computational Drafting of Plot Structures for Russian Folk Tales. *Cognitive Computation*, 07/2015 2015.
5. Pablo Gervás, Carlos León, and Gonzalo Méndez. Schemas for narrative generation mined from existing descriptions of plot. In *Computational Models of Narrative*, Atlanta, Georgia, USA, 05/2015 2015. Scholoss Dagstuhl OpenAccess Series in Informatics (OASIcs), Scholoss Dagstuhl OpenAccess Series in Informatics (OASIcs).

6. Raquel Hervás and Pablo Gervás. Case-based reasoning for knowledge-intensive template selection during text generation. In Thomas R. Roth-Berghofer, Mehmet H. Göker, and H.Altay Güvenir, editors, *Advances in Case-Based Reasoning*, volume 4106 of *Lecture Notes in Computer Science*, pages 151–165. Springer Berlin Heidelberg, 2006.
7. Gilbert Müller and Ralph Bergmann. Compositional Adaptation of Cooking Recipes using Workflow Streams. In *Computer Cooking Contest, Workshop Proceedings ICCBR 2014*, Springer, 2014. The original publication is available at www.springerlink.com.
8. Santiago Ontañón and Enric Plaza. Amalgams: A formal approach for combining multiple case solutions. In Isabelle Bichindaritz and Stefania Montani, editors, *Case-Based Reasoning. Research and Development*, volume 6176 of *Lecture Notes in Computer Science*, pages 257–271. Springer Berlin Heidelberg, 2010.
9. Santiago Ontañón and Jichen Zhu. On the role of domain knowledge in analogy-based story generation. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Two*, IJCAI'11, pages 1717–1722. AAAI Press, 2011.
10. F. Peinado and P. Gervás. Evaluation of automatic generation of basic stories. *New Generation Computing*, 24(3):289–302, 2006.
11. Federico Peinado. *Un Armazón para el Desarrollo de Aplicaciones de Narración Automática basado en Componentes Ontológicos Reutilizables*. PhD thesis, Universidad Complutense de Madrid, Madrid, 2008.
12. Rafael Pérez y Pérez and Mike Sharples. MEXICA: A computer model of a cognitive account of creative writing. *Journal of Experimental & Theoretical Artificial Intelligence*, 13(2):119–139, 2001.
13. Vladimir Propp. *Morphology of the Folk Tale*. Akademija, Leningrad, 1928.
14. M. Riedl and Carlos León. Toward vignette-based story generation for drama management systems. In *Workshop on Integrating Technologies for Interactive Stories - 2nd International Conference on Intelligent Technologies for Interactive Entertainment*, 2008.
15. Mark Riedl and Carlos León. Generating story analogues. In *AIIDE*, 2009.
16. Mark O. Riedl and Neha Sugandh. Story planning with vignettes: Toward overcoming the content production bottleneck. In *Interactive Storytelling*, volume 5334 of *Lecture Notes in Computer Science*, pages 168–179. Springer, 2008.
17. R. Schank. *Dynamic Memory : A Theory of Reminding and Learning in Computers and People*. Cambridge University Press, 1982.
18. R. Schank and R. Abelson. *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures*. L. Erlbaum, Hillsdale, NJ, 1977.
19. Scott Turner. *MINSTREL: A Computer Model of Creativity and Storytelling*. PhD thesis, University of California at Los Angeles, Los Angeles, CA, USA, 1992.
20. Wolfgang Wilke and Ralph Bergmann. Techniques and knowledge used for adaptation during case-based problem solving. In *Proceedings of the 11th International Conference on Industrial and Engineering Applications of Artificial In Telligence and Expert Systems: Tasks and Methods in Applied Artificial Intelligence*, IEA/AIE '98, pages 497–506, London, UK, UK, 1998. Springer-Verlag.

Seeking Divisions of Domains on Semantic Networks by Evolutionary Bridging

João Gonçalves, Pedro Martins, António Cruz, and Amílcar Cardoso

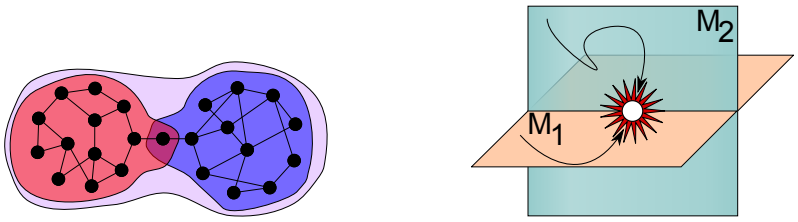
CISUC, Department of Informatics Engineering, University of Coimbra
{jcgonc,pjmm}@dei.uc.pt
antonioc@student.dei.uc.pt
amilcar@dei.uc.pt

Abstract. Computational Creativity systems based on Conceptual Blending (CB) and Bisociation theories operate on input knowledge to reveal seemingly unrelated information. The input spaces or domains can be of various sources and contain vast amounts of knowledge. It is central a process that selects useful building blocks of semantic data that does not narrow the search space of the creative algorithm. It is also vital that the data selection process is of high performance in order to handle a large knowledge base in a useful time. With those objectives in mind, we propose an evolutionary high performance algorithm that extracts two semantic sub-graphs from a knowledge base to be used as building blocks in computational blending processes.

1 Introduction

A creative process can be seen as a form of heuristic search for a construct on a vast semantic space of concepts and domains. In this paper we propose an evolutionary approach inspired by the work of Nagel [6] for selecting two domains from a broader knowledge structure using high performance algorithms. This allows a faster extraction of a more concise representation of the data to be used in computational concept generation techniques, such as CB and Bisociative Knowledge Discovery, among other applications. As the amount of available knowledge dramatically expands each year, high performance algorithms are required to cope with the extraction of new insights, together with growth of multidisciplinary knowledge bases. In [3] Juršič, based on the ABC model by Swanson [11] [10], proposes the CrossBee system for supporting creativity insight in knowledge discovery of literature. In the ABC model, Swanson remarks that reference citations and other bibliographic indications potentially reveal new knowledge, which is not clearly intended neither logically exposed in the literature. That is, ABC exposes the $A \implies B \implies C$ logical consequence, being B the term which relates the remaining terms A and C. Using this idea, CrossBee tries to explore bridging terms linking two apparently disconnected literature domains. In CrossBee, the bridge terms contain relations between two terms, each from a different domain, that were mined from a literature knowledge base. The terms are extracted from various sections in the literature texts, such as bibliography, citations, logical consequences and other references present in the texts. Having the literature containing the terms from A referencing terms regarding B, and

simultaneously the literature C also references terms from B, then a unintended modus ponens inference suggest a hidden relation between A and C. Hence, the system following closely the ABC idea by Swanson.



(a) (b)

Fig. 1: Two domains connected by a single concept (left: adapted from [1]) and the juxtaposition of two frames of reference in Koestler's Bisociation (right: adapted from [4]).

A similar work by Nagel et al. [6] introduces a formalised spreading activation algorithm to identify bridging concepts in a semantic graph (Fig. 1a). The bridging terms interconnect nodes from disjoint semantic domains, following an identical idea to CrossBee. However, the main intention in this case is to juxtapose two apparently unrelated domains through a single term [5]. This notion of pairing two disjoint frames of reference using a singular connection was put forth by Koestler and named Bisociation [4] in his work, *The Act of Creation* [4]. There, Arthur Koestler attempts to describe creative behaviour present in humour, arts and science. This model consists on "the perceiving of a situation or idea ... in two self-consistent but habitually incompatible frames of reference (M_1 and M_2)" as shown in Fig. 1b. For instance, in humour, bisociations could relate the unforeseen transformation from one meaning to another [7]. In their paper, Nagel explores a search space, defined using a bisociation score, which rates individually each bridging node subdividing the semantic graph in two completely disjoint sets. However, their approach does not allow a tolerance of the intersection between the two domains. Thus, it is in a sense a hard margin solution and in our opinion, it terminates the search prematurely in real world problems, as underlined in their conclusion. However, their highly formalised work served as a basis for our present approach. In the following section, we offer our evolutionary approach in the form of a Genetic Algorithm (GA), inspired by the work of Nagel.

2 Algorithm

The purpose of the algorithm is to identify two partially overlapping sub-graphs S_0 and S_1 of a larger semantic graph S. Given their structure, interrelations and arrangement within the larger graph, we believe the sub-graphs could be seen as domains of knowledge in the broader semantic graph.

The cardinality of each graph structure is identified by the symbol #. Thus, $\#S$ is the number of nodes existing in the graph S and we denote this quantity as the size of the graph. Each sub-graph represents a network of highly interconnected nodes, which if belonging to a semantic graph, could represent a domain of related concepts [1]. Both sub-graphs share at least a single node N_b, the bridge node, and the sub-graphs should be balanced [6] regarding a split through the bridge node. The size of both sub-graphs $\#S_0$ and $\#S_1$ should be maximized, with only the condition of $S_0 \cap S_1 = \{N_b\}$. Then, a unique path will flow from one sub-graph to the other through the bridge node which has the unique index $b \in \{1 \dots \#S\}$. When this happens, the unique bridge node may represent a possible bisociation which juxtaposes one domain (sub-graph) into the other (Fig. 1b).

A degree d_i of a node N_i with $i \in \{1 \dots \#S\}$ represents the number of incident edges to that given node. Nodes with $d = 2$ represent a single relationship between two concepts, being these the most likely candidates for a bridge node [6]. The reasoning behind this choice is the interest in mapping two domains over a single and clear semantic relationship, through the bridge node. In this case, any concept from one sub-graph can be projected onto any other concept from the other sub-graph, offering a foundation for further transformations of concepts using processes from bisociation and CB [7].

Otherwise nodes with $d \geq 3$ map a more vague set of relations between connected concepts. Intuitively, the view of an idea in two distinctly but opposing views is more fine tuned to two set of concepts (two domains) connected by a single node [6]. A simple example which demonstrates this idea is seen in Fig. 1a. On the other hand, highly interconnected nodes express a deeply related network of information or domain. Using the above criteria, the discrete function which rates the optimality (fitness) of the bridge node N_b is defined in (1):

$$f(S_0, S_1, d_b) = \begin{cases} \frac{1}{\alpha^{\frac{|\#S_0 - \#S_1|}{\#S_0 + \#S_1}} + 1} \cdot \log(\#S_0 + \#S_1) \cdot 2^{-\beta(d_b - 2)}, & \text{if } d_b \geq 2 \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

The fitness function receives as arguments the sub-graphs S_0, S_1 and the degree of the bridge node N_b as the variable d_b. The parameter α controls how similar in size the sub-graphs S_0 and S_1 are required to be, with increasing α exhibiting greater size similarity. The parameter β is used to control the penalisation given to bridge nodes with a degree $d > 2$, with the penalisation exponentially proportional to the value of β. If the degree of node being rated is 1, that is, a terminal node with a single relation, then f is set to 0 in order to prevent the GA to select terminal nodes as bridge.

Globally, a GA evolves a population of chromosomes where each chromosome represents a bridge node and two sub-graphs of the initial semantic network. Each time a new individual is created with a given bridge node, a breadth first search is executed starting in the latter node and into neighbouring nodes. The dual diffusion process (a sort of spreading activation) progresses radially until a given expansion depth is reached when both sub-graphs intersect, or all the

nodes in the network have been explored. From this expansion, we calculate a performance score representing this graph division which will represent the chromosome in the global fitness landscape. Thus, the GA evolves subdivisions of the semantic network aiming to find the bridging node that maximises the size of the sub-graphs found.

A GA was chosen in order to relax the search for global optima. Given a specific graph, the search space can be of very high complexity which is likely when handling large semantic networks. The optimisation task is represented by the fitness function and the genetic operators embody the transformations that move the bridge node through the semantic graph. By using a GA, it is easy to parallelise the search algorithm with the following steps:

- fitness functions are evaluated in parallel threads;
- new individuals which will define the next generation are created in parallel threads;
- the generation of random chromosomes for the initial population and when the GA stagnates is executed in parallel threads.

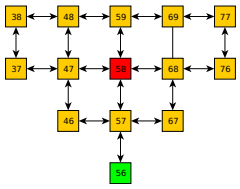


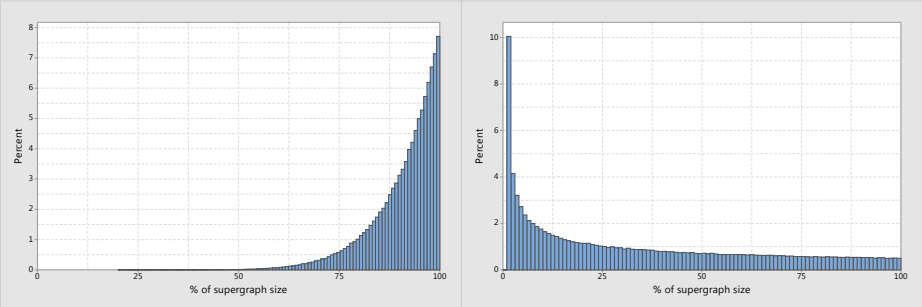
Fig. 2: A combination of random translation movements (a random walk) starting at the red node and given time, eventually leads to the green node. Best viewed in colour.

In our GA we use entirely the mutation operator. Using Fig. 2 as an example, it can be seen that the fundamental operation required to traverse the search space (the graph) is the translation of the bridge node. Thus, the search algorithm samples stochastically (walks randomly) the locations in the graph that maximise the fitness function. The mutation changes the location of the bridge node using the connected nodes (neighbourhood) recursively. In order to allow the bridge node to jump to distant sections of the graph, we use a quadratic random number generator. The probability function controlling the locality behaviour of this process decreases with the increasing number of jumps the bridge node is allowed to make. This number is given in (2).

$$j(r, s) = r^{\gamma} s + 1 \tag{2}$$

The parameter $r \in [0, 1[$ represents a previous output of an uniform pseudo random number generator. The number of nodes $\#S$ in the graph S is represented by the parameter s. The parameter $\gamma \in]0, +\infty[$ controls the average jump distance (translation deepness) the mutation applies to the bridge node.

If $\gamma >> 1$, the mutation tends to move the bridge node towards nearby nodes (Fig. 3b). When $\gamma \to +\infty$, the translation tends to move each bridge node to



(a) $\gamma = 0.1$ (b) $\gamma = 2$

Fig. 3: Probability Density Functions for two values of γ when using the bridge jump mutation function (2).

one of the connected neighbours. For $\gamma \in]0, 1[$, particularly nearby 0, tends to translate the bridge node to distantly connected nodes (Fig. 3a) and, in a sense, promote a random search of the fitness landscape. The case where $\gamma = 1$ forces a straightforward stochastic search whereby each bridge node can be moved to any other node position of the domain S with constant probability.

After the mutation has been applied to the chromosome, the expansion of sub-graphs from the full domain is executed again from the newly calculated bridge node. When this completes, two new sub-graphs divide part (or all) of the domain with roots at the bridge node and depth $d_{(0...1)}$. This expansion is a type of breadth first search starting at the node N_b so that the first nodes to explore are the nearby nodes. The main idea behind this reasoning is shown in Algorithm 1. Using two pairs of open (to expand) and closed (already expanded) nodes, the algorithm inserts the visited nodes in the two sub-graphs which will represent the sub-domains S_0 and S_1. An example of the process is seen in (Fig. 5) from where two different coloured sub-graphs emerge (Fig. 5). The function `nodes`(S_i) returns the set of nodes $\{N_i\}, i \in \{0, \dots \#S_i - 1\}$ in the sub-graph $\{N_k\}, k \in \{0, 1\}$. The variables O_i and C_i represent respectively the set of open (to visit) and closed (already visited) nodes related to the sub-graph i. The function `split`(N_b, S) divides the neighbourhood of the bridge node N_b in two sets of nodes as evenly as possible. When the neighbourhood of N_b is odd, a randomly chosen set S_i receives the additional node so that in the worst case, the difference of cardinality between S_0 and S_1 is 1.

The function `expandOneLevel`(S_i, O_i, C_i, S) cycles through all the nodes in the set O_i, inserts each visited node in the set C_i and in the sub-graph S_i, including the connected edges. Then it extracts the neighbourhood of each visited node and inserts the neighbour nodes in the set O_i, so that in the next iteration of the function `createSubgraphs`() the algorithm expands from the current neighbourhood of O_i. Thus, the function `expandOneLevel()` executes an equivalent single iteration of a breadth first search at the same deepness level. All the nodes and edges are obtained from the graph S.

Algorithm 1 Function createSubgraphs()

function CREATESUBGRAPHS(N_b, S)
  $\{S_0, S_1\} \leftarrow$ `split`(N_b, S)
  $O_0 \leftarrow$ `nodes`(S_0)
  $O_1 \leftarrow$ `nodes`(S_1)
  $C_0 \leftarrow N_b$
  $C_1 \leftarrow N_b$
  $I \leftarrow \emptyset$
  repeat
    `expandOneLevel`(S_0, O_0, C_0, S)
    `expandOneLevel`(S_1, O_1, C_1, S)
    $I \leftarrow S_0 \cap S_1$
  until $\frac{\#I}{\#S_0+\#S_1} \geq \tau \vee \#S_0 = 0 \vee \#S_1 = 0$
end function

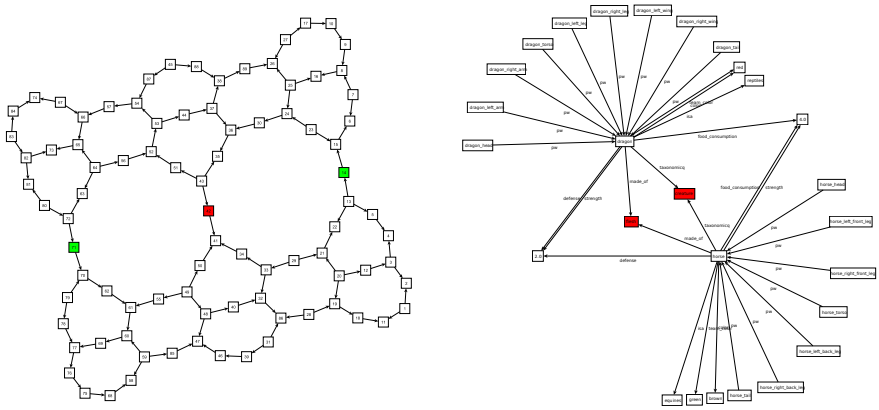
Starting at the bridge node N_b the expansion grows radially throughout the connected nodes, creating the sub-graphs while visiting the explored nodes until the sub-graphs intersect. For a graph in structure similar to Fig. 1a, the sub-graphs are expected to intersect only in the bridge node. However, in real cases, while the expansion is taking place, the intersection can suddenly show a small amount of nodes when in comparison with the size of both sub-graphs S_0 and S_1. When this happens, the algorithm may not be able to find a clean (and useful) division of the graph S.

Using a similar idea to Soft Margin in [2], we include the parameter $\tau \in \mathbb{R}_0^+$ to allow more than one bridge node connecting the sub-graphs S_0 and S_1. With $\tau = 0$, the intersection I between the sub-graphs is allowed to contain only the bridge node, as in Nagel. When the ratio of the intersection to both sub-graphs size increases above τ, the algorithm stops and returns the most recently created sub-graphs starting at node N_b. In sum, τ represents the trade-off between the penalization of highly interconnected sub-graphs and the maximisation of the size of those sub-graphs.

Consider the following example: after a 5 level expansion, the intersection of the two sub-graphs with a size of 2000 nodes each, suddenly increases from 1 (the bridge node only) to 100. This means that the fifth iteration raised the sub-graphs size to intersection ratio from $\frac{1}{2000+2000} = 0.025\%$ to $\frac{100}{2000+2000} = 2.5\%$, a $100\times$ fold increase. It may happen that the sub-graphs contain useful knowledge and for this reason, they should not be discriminated. Depending on the situation at hand, the parameter τ chosen to control the ratio may or not be significant.

3 Results and discussion

The feasibility of our algorithm was tested using three semantic graphs. The first, shown in Fig. 4a, was generated exclusively to test the theory supporting the algorithm. It contains 89 nodes and 106 unlabelled directed edges. The second is from the Horse-Dragon experiment, a well known semantic graph in Conceptual



(a) 89 node graph. (b) Horse-Dragon graph.

Fig. 4: Structure of the 89 node and Horse-Dragon graphs. The highest rated nodes are shown coloured. Best viewed in colour.

Blending, supplied by the authors of [8]. This semantic graph contains 32 relations between 32 attributes of the animals horse and dragon, such as physical parts, health resistance and some taxonomic properties (Fig. 4b). The last is the Perception semantic graph from [9]. The author of Perception defines his knowledge base as a summary of manually annotated common sense concepts and their relations. It contains 3892 nodes and 345463 edges. Unless otherwise stated, the parameters used for the graph division algorithm were $\tau = 0$, $\alpha = 4$, $\beta = 4$ and $\gamma = 2$. The GA evolved a population of 10^3 chromosomes with a mutation rate of 100%, no crossover and a maximum number of 10^3 evolved generations.

Before the experiments, we validated the algorithm with a 111 node graph (Fig. 5) containing 188 directed edges. After the conclusion of the GA, the two sub-graphs S_0 (green) and S_1 (cyan) are juxtaposed through the bridge node with the label 56. The GA stopped when the intersection between the sub-graphs included the nodes labelled 17, 95 and the bridge node with label 56. Afterwards, we proceeded with the experiments on the three semantic graphs.

Table 1: Fitness scores f for the four highest rated bridge nodes of the *Horse-Dragon* semantic graph.

f	degree(N_b)	label(N_b)	$\#S_0$	$\#S_1$
3.989	2	creature	15	15
3.989	2	flesh	15	15
0.249	3	4	15	15
0.249	3	2	15	15

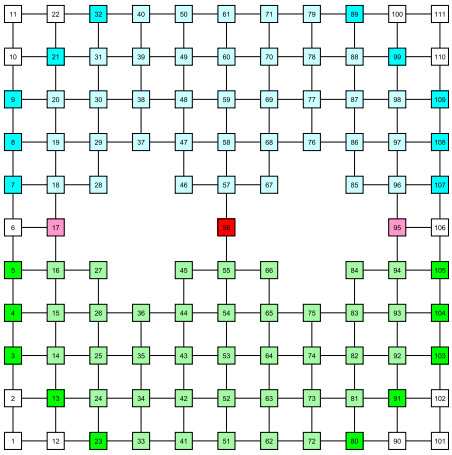


Fig. 5: Optimal sub-graph configuration of the 111 node test graph showing bridge node (red), intersection nodes (pink) and the two created sub-domains (green and cyan) each with a depth of 8 nodes starting at the bridge node. Calculated with intersection tolerance $\tau = 0$. Best viewed in colour.

Our algorithm reported a high amount of possible bridge nodes in the 89 node graph (Fig. 4a). From those, the 3 highest scored nodes are shown coloured, where the node in red scored 50% higher than the green nodes.

The Horse-Dragon [8] semantic graph is shown in Fig. 4b with the four highest rated chromosomes presented in Table 1. The majority of the nodes are terminal ($d = 1$) where a small number of highly interconnected nodes ($d \geq 2$) are clearly visible (Fig. 4b) labelled as *horse* and *dragon*. The best chromosomes generated by the GA produced were the two pairs of sub-graphs with each pair linked by the nodes with label *creature* and *flesh*.

In order to study our algorithm with a more complex and practical problem, we researched the Perception [9] knowledge base with two experiments. For the first, we did a study regarding the effect of τ in the size of the two sub-graphs. As shown in Table 2, the parameter τ highly influences the size of both sub-graphs. Having the Perception graph 3892 nodes, for certain τ values, one or both of the sub-graphs contain more than half the nodes from Perception. Therefore, a compromise has to be made so that both sub-graphs do not drastically intersect between themselves. However, both should contain a minimum amount of knowledge and relations to be useful for CB and Bisociative Knowledge Discovery. From Table 2, an interesting improvement in the size of the sub-graphs happens when τ changes from 0.05 to 0.1. With $\tau \geq 0.5$ the fitness function f does not increase, implying that the limit of the graph has been reached as the size of both sub-graphs are equal and maximum.

For the second experiment, we set $\tau = 0.1$ in order to limit the intersection between the two sub-graphs to 10% of their combined size. A list of results is present in Table 3, with all the bridge nodes having degree of 2. The fitness

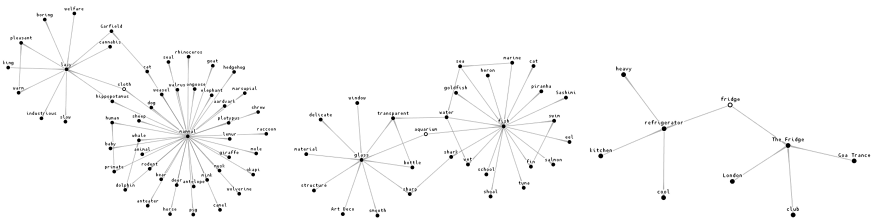
Table 2: Fitness scores f of the highest rated bridge nodes for the Perception semantic graph when varying τ.

τ	f	degree(N_b)	label(N_b)	$\#S_0$	$\#S_1$	$\#(S_0 \cap S_1)$
0.00	4.35	2	panther	47	46	0
0.05	7.07	2	Jerry Springer	586	586	7
0.10	7.70	2	Times	1919	1991	98
0.15	8.19	2	Jesus Christ	2173	2199	119
0.20	8.27	2	Pulp Fiction	2121	2132	186
0.25	8.30	2	wrestling	2264	2281	291
0.33	8.63	2	ashes	3039	3054	943
0.50	8.95	2	emerald	3868	3868	3868
0.75	8.95	2	chromosome	3868	3868	3868

Table 3: Fitness score f for 22 bridge nodes, from the Perception semantic graph with $\tau = 0.1$.

f	degree(N_b)	label(N_b)	$\#S_0$	$\#S_1$	$\#(S_0 \cap S_1)$
7.70	2	Times	1919	1991	98
7.52	2	Athens	1474	1522	62
7.07	2	Jerry Springer	586	586	7
7.03	2	sloth	1242	1177	32
6.98	2	herd	714	729	4
6.96	2	fox	984	1031	27
6.78	2	pilot	529	522	11
5.79	2	aquarium	949	820	13
4.99	2	fridge	427	514	1

declines with the increasing unbalancing between the sub-graphs S_0 and S_1. In Figs (6) we show some of the structures of the nearby connections. It is interesting to observe the relations between connected domains for the "sloth", "aquarium" and "fridge" nodes. We find the last case peculiar, as we did not knew of a heavy and cool (or cold?) trance band. For the remaining bridge nodes, we leave their insight to the reader's judgement.



(a) "sloth" bridge node. (b) "aquarium" bridge node. (c) "fridge" bridge node.

Fig. 6: Three examples of bridging nodes and their neighborhood present in Table 3.

4 Conclusions

In this work we proposed an evolutionary approach to support computational concept generation systems and knowledge discovery. Building on the work of Nagel [6], our work allows the discovery of knowledge divisions in large semantic graphs and the identification of possible key concepts which interconnect the sub-graphs. The algorithm supports various parameters to fine tune this division process in accordance with real world knowledge bases, so that different relations between knowledge domains can be researched and hopefully, give possibility to

new insights between those domains. Lastly, by using a high performance algorithm, the exploratory process can be done in useful time. In the future we expect to improve our approach by experimenting with graph similarity. It would also be interesting to use a form of feedback loop by integrating a concept generating algorithm. This way, the system would direct its search towards bridging nodes and sub-graphs that would be more useful to the task that follows the GA.

Acknowledgements. The authors acknowledge the financial support from the iCIS (Intelligent Computing in the Internet of Services) project (CENTRO-07-0224-FEDER-002003).

References

1. Michael R. Berthold, editor. *Bisociative Knowledge Discovery*. Volume 7250 of Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2012.
2. Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.
3. Matjaž Juršič, Bojan Cestnik, Tanja Urbančič, and Nada Lavrač. Cross-domain literature mining: Finding bridging concepts with crossbee. In *Proceedings of the 3rd International Conference on Computational Creativity*, pages 33–40, 2012.
4. Arthur Koestler. *The Act of Creation*. New York:Macmillan, 1964.
5. Tobias Kötter, Kilian Thiel, and Michael R Berthold. Domain bridging associations support creativity. 2010.
6. Uwe Nagel, Kilian Thiel, Tobias Kötter, Dawid Piatek, and Michael R. Berthold. Towards discovery of subgraph bisociations. In Michael R. Berthold, editor, *Bisociative Knowledge Discovery*, volume 7250 of *Lecture Notes in Computer Science*, pages 263–284. Springer Berlin Heidelberg, 2012.
7. F Pereira. *Creativity and artificial intelligence: a conceptual blending approach*. Berlin: Mouton de Gruyter, 2007.
8. Paulo Ribeiro, Francisco C. Pereira, Bruno Marques, Bruno Leitão, and Amílcar Cardoso. A model for creativity in creature generation. In *Proceedings of the 4^{th} Conference on Games Development (GAME ON'03)*. EuroSIS / University of Wolverhampton, 2003.
9. Tom De Smedt. *Modeling Creativity: Case Studies in Python*. Uitgeverij UPA University Press Antwerp, 2013.
10. Don R Swanson. Fish oil, raynaud's syndrome, and undiscovered public knowledge. *Perspectives in biology and medicine*, 30(1):7–18, 1986.
11. Don R Swanson. Two medical literatures that are logically but not bibliographically connected. *Journal of the American Society for Information Science*, 38(4):228–233, 1987.

Case-Based Slogan Production

Martin Žnidaršič[1,3], Polona Tomašič[2,3], and Gregor Papa[1,3]

[1] Jožef Stefan Institute, Jamova cesta 39, SI-1000 Ljubljana, Slovenia,
[2] OLAII d.o.o., Pot za Brdom 100, SI-1000 Ljubljana, Slovenia
[3] Jožef Stefan International Postgraduate School, Jamova cesta 39, SI-1000 Ljubljana, Slovenia
martin.znidarsic@ijs.si
polona.tomasic@gmail.com
gregor.papa@ijs.si

Abstract. This paper presents a case-based approach to automated generation of slogans. We use a collection of cases out of which the selected ones get transformed and adapted to a new context that is represented by a textual description of the slogan's target. We also propose a methodology for evaluation and ranking of the final results. The approach is experimentally applied to two real-world use cases. The results indicate the ability of the approach to create slogan prototypes and reveal the issues to tackle in the next steps of solving this challenging problem.

Keywords: slogan generation, CBR, transformational adaptation, computational creativity, natural language

1 Introduction

Invention of slogans is a task that demands knowledge about the object of the slogan, its context and the intended message. However, such knowledge is not enough, as it has to be used in a creative way to produce a slogan that is novel, interesting and memorable. As a task that demands common knowledge and a high level of creativity, slogan generation is inherently difficult to automate. The aim of the work presented in this paper is to contribute to solutions of this challenging problem.

Our approach uses the texts of slogan cases to create new slogans that follow the grammatical structure of the initial cases, but use different words and phrases that are related to the slogans' target objects and contexts. As we use a collection of cases that we build upon and transform, this approach can be considered an application of case-based reasoning (CBR) in the domain of computational creativity.

It is very hard to automatically generate novel slogans that would be ready for use without further adaptations and corrections. This is not the case for simple template-based techniques[4], but these are not useful for our purposes

[4] Such as: "X, you have to buy it!" (put the name of the product in place of X).

as despite producing ready-made solutions, they are not innovative and do not produce context dependent results.

The outputs of the more innovative approaches often contain grammatical errors and semantic incoherencies. These outcomes can be considered slogan prototypes rather than slogans. They are useful in the conceptualization phase, as an addition to other techniques for production of solution drafts.

The case-based slogan generation is an example of a hybrid approach: it uses case texts, but not as rigid templates and it aims at incorporating some of the context of the slogan's target object. We have experimentally applied this methodology to two use cases. The relevant results and their assessments are provided in the paper, along with a discussion of the strong and weak points of our approach.

2 Related Work

Automatic generation of innovative creative artefacts that have a defined semantics is very challenging and the outcomes of such systems and methods are usually not ready for use without some sort of human curation. The computational creativity problems that are similar to slogan generation in terms of difficulty and representation are generation of jokes [1, 10], poems [4, 3] and generation of stories [2, 6], to some extent also the automatic generation of acronyms [11].

In the case of automated generation of slogans, there are only two lines of research work to the best of our knowledge: (I) the BrainSup approach by Özbal et. al. [9], which is the most well known and (II) the work by Tomašič et. al. [12], which is heavily influenced by the BrainSup approach, but complements it with the use of a genetic algorithm and additional evaluation functions. While the former expects relevant meta-data to be provided by the user, such as the keywords, the domain, etc., the latter is made to be completely autonomous. Consequently the reported results of BrainSup are of much higher quality.

In terms of CBR, the studies related to the work in this paper are the ones that are concerned with the use of textual data in CBR [13, 8]. Among these, we can also find some that are related by domain, such as the study on the use of CBR for story generation [5].

3 Slogan Collection

In our experiments we used a manually generated dataset of 5183 distinct items, each containing words transformed to lowercase, that appear in an example of a slogan.

Besides the words with their grammatical characteristics, we do not store other information, for example the particular product or product type that the slogan might be used for. Most of the slogans are used for promotion of the values and characteristics of a company and all its products, which might be numerous and diverse. As the characteristics of the products are reflected in the

characteristics of the company and vice versa, it is usually difficult to determine whether a slogan is meant to be general or product-specific.

Cases in the collection are targeting diverse products and companies, from housing and financial services to food and cosmetics products. They differ a lot also in other characteristics, like length for example. The shortest one in the collection is only a 4 characters long word, while the longest one consists of 215 characters and 34 words. The median number of characters in the cases of this collection is 28, while the median number of words is 5.

4 Generation Process

The slogan generation process mostly follows the usual CBR steps [7] and is also presented in this fashion, by first describing the *retrieval* of similar cases, then the *adaptation and transformation* to suit a particular target and finally the *evaluation* and ranking of results.

4.1 Retrieval of Relevant Cases

Retrieval of cases that are relevant for a given problem is not trivial in our setting. Namely, the only input into our system is a textual description of the target (a company or a product), while our knowledge base consists of exemplary texts. In the absence of meta-data, which would, ideally, describe the context of the slogan and its target, we use only the textual information of the slogan examples and the target's textual description.

The retrieval process consists of two steps: (I) preprocessing of textual representations and (II) selection, based on similarity of words. First, the text of each slogan and the textual description of the slogan's target is transformed into a bag of words representation from which all the stopwords are removed (we have used the nltk library[5] for this purpose) and all the characters are transformed to lower case. Then, the items in the case-base are selected for adaptation, based on the matching of their words with the words in the target's description. If an item contains a word that appears also in the description of the target, it gets added to the collection of relevant cases. If it matches the target text in n words, it gets added n-times. We can describe this with the following equation:

$$n = |W_s \cap W_t|, \tag{1}$$

where the number of copies of an item in the collection of relevant cases (n) is expressed as the cardinality of the intersection among the words from the slogan (W_s) and the words from the target's description (W_t). If the intersection is empty, the particular item does not get added to the collection of relevant cases.

This way, the slogans with more words that appear also in the target's text have more instances in the collection of selected cases and consequently more of their (diverse) transformations represented in the final results.

[5] http://www.nltk.org/

4.2 Transformation

The selected items are transformed by insertion of words from the target's description. For each selected case-base item we exchange each of its words with probability p. Such a word gets exchanged with a randomly selected word from the target's description that has a matching part of speech (POS) tag while the punctuation marks are left unchanged. This way, repeated items that appear in the selection get transformed differently, as the exchanged words are in general different and their replacements are usually also different.

The exchange probability parameter p controls the level of diversity of the transformed items from the initial ones. Low values of p cause the resulting slogans to be more similar to their initial cases, thus they are less innovative and can be seen as imitations. High values of p on the other hand, cause the resulting slogans to be more novel, better connected to the target domain, but also more uncontrolled, with a higher frequency of grammatical errors and semantic incoherencies. As we prefer the results of the latter kind, we used $p = 0.75$ in our experiments.

4.3 Evaluation and Refinement

Due to the generation procedure, the transformed items often (depending on the parameter p) contain grammatical and semantic errors. To assess the results in this respect and to alleviate this problem, the outputs get evaluated and the final results of our approach are presented in a descending order of their evaluation scores.

For the purpose of evaluation, we represent each transformed item as a multiset[6] or a bag B_{ts} of bi-grams. For example:

```
you just have to buy this to be happy.
```

would be represented as:

```
{(you, just), (just, have), (have,  to), (to,  buy), (buy, this),
(this, to), (to, be), (be, happy)}.
```

Likewise, we create a multiset B of all the bi-grams that appear in all the examples in our case base and the input target text.

Each transformed item is then scored according to the number of its bi-grams from B_{ts} that appear also in B. This way, the results that have more bi-grams that appear in related texts (all the exemplary texts and the target's text) are scored higher. We expect that such results are constructed in a more meaningful way, at least locally in a word-to-word sense. However, by considering only the number of the matching bi-grams, the evaluation would be biased towards longer, and not necessarily more meaningful slogans. Therefore, our evaluation score S

[6] Namely, we want to allow a bi-gram to appear multiple times in our collection.

is a ratio of the number of the matching bi-grams and the number of all the bi-grams of the evaluated slogan:

$$S = \frac{|B_{ts} \cap B|}{|B_{ts}|} \ . \tag{2}$$

The final output of our approach are therefore the transformed selected slogans, ordered according to S.

5 Experiments

The approach presented in Section 4 was applied to two exemplary use cases: companies Sentinel[7] and Olaii[8]. Sentinel provides solutions for monitoring of a state of a boat or a fleet of boats, while Olaii is providing a system for payments and access management for events.

The input textual descriptions in both use cases were very raw, as we used all the text from their respective home pages, together with the boilerplate text such as the menu items, disclaimers, etc. The inputs were intentionally not cleaned in order to get an assessment of results from a very straightforward and realistic kind of use.

The first 10 and the last 10 results for Sentinel and Olaii are shown in Tables 1 and 2, respectively. According to our qualitative assessment, the outputs with the top ranks are clearly of higher quality than the bottom ranked ones, while the quality of the outputs is generally too low for practical use.

We have also experimented with the use of lower and higher values of the parameter p. As its impact is not very profound and can be observed only when one inspects a large number of outputs, we do not present these results here. Among the badly ranked outputs, as expected, the ones obtained with lower values of p (for example 0.50) are usually more readable and grammatically correct and the ones obtained with high values of p (like 0.90) are worse in this respect. Among the highly ranked outputs, lower values of p cause more results similar to initial ones to appear among the outputs, while the quality is not affected much even with the use of high values of p. This is most probably due to the evaluation and ranking procedure, which penalizes grammatically incorrect and incoherent slogans. The more abundant erroneous outputs that are expected to be produced with high values of p are thus prevented from appearing among the well ranked results. Therefore, it seems that it is sensible to use large values of p as this ensures production of less outputs that are similar to the already existing ones, while the evaluation and ranking prevents the comparatively larger amount of erroneous solutions to be present among the top results - the ones that are of interest in practice.

[7] http://www.sentinel.hr/
[8] http://cashless.olaii.com/

Table 1. Best and worst scored slogans that were generated for the Sentinel use case. The slogans with an equal bi-gram ratio score S are sorted according to the number of words (shortest first). An asterisk (*) is put in places where product names appear in the transformed slogans. Outputs that by chance match an initial item are removed from the ranked list.

Rank	Generated slogan	S
1	immediately what you need to be your best.	0.750
2	you enjoy our promises to you.	0.667
3	go * and warn the driving to you!	0.625
4	the one and only possible.	0.600
5	free enterprise with every issue.	0.600
6	simple boat to like you	0.600
7	an your security needs under one vacation.	0.571
8	you enjoy clearly when you enjoy it.	0.571
9	at the men in charge about eye.	0.571
10	battery you need from conception to reception.	0.571
...	...	...
338	a wholesome anchor with yet or detection.	0.000
339	a system alerts only , it clearly receives.	0.000
340	a alert is voltage holidays at us!	0.000
341	a most possible anchor need before all boat.	0.000
342	only it 're going , it enjoy activating immediately.	0.000
343	your leave , information provides reliable , be at times..	0.000
344	sensors batteries you provides losing , and going , or sentinel.	0.000
345	sensors batteries you notifies going , and going , or sentinel.	0.000
346	entering healthy batteries about one eye , over all worries.	0.000
347	gps enjoy about , and they do away be out!	0.000

6 Discussion and Conclusion

The case-based generation of slogans is an approach that uses information from examples of solutions and aims at transforming them with regard to a target entity context into new slogans. Our method allows setting a parameter that controls the expected level of distortion of the original solution and adaptation to the target entity.

Our experiments indicate that the CBR-based approach can create artefacts, which can be used as prototype solutions for further (manual or automatic) refinement. Outputs of some experimental runs even produced good original slogans that could be used without further modification, such as:

`the most reliable anchor of your solution.`

which appeared among top ranked outputs with $p = 0.90$ for the Sentinel case. However, the experiments also show that the approach often results in erroneous and even meaningless solutions and that in general the amount of such noise (at the values of the distortion parameter that allow innovative slogans) is

Table 2. Best and worst scored slogans that were generated for the Olaii use case. The slogans with an equal bi-gram ratio score S are sorted according to the number of words (shortest first). All examples that are ranked 10 and have the same number of words are presented. An asterisk (*) is put in places where product names appear in the transformed slogans. Outputs that by chance match an initial item are removed from the ranked list.

Rank	Generated slogan	S
1	the best value of the event.	0.833
2	get to become a world.	0.800
3	the way you should have.	0.800
4	you find your visitors.	0.750
5	do you know you?	0.750
6	on all everything is a story to handle.	0.750
7	the first time is up the best.	0.714
8	all the * you are to reduce.	0.714
9	you can top-up the party to you.	0.714
10	who you find is what you are.	0.714
...	...	...
2136	an necessary few animal.	0.000
2137	benefits hard , once n't.	0.000
2138	more alerts , less habits.	0.000
2139	less visitors , less stations.	0.000
2140	a digital control to again.	0.000
2141	the product cards/wristbands are !	0.000
2142	you 're controlling better via n't.	0.000
2143	it will manage more good per you.	0.000
2144	them will steal the deeper you again.	0.000
2145	you better transfer more , you deposit Do.	0.000

substantial and further improvements are needed in order for the method to be applicable in practice.

A positive indication of the experimental results is the performance of the evaluation method, which seems to be useful, according to qualitative analysis of the result ranking. This is an encouraging result, as evaluation represents a big challenge in the problem domains of computational creativity. However, to strengthen this indication, which is currently supported only by the qualitative observations by the authors, a more elaborate evaluation procedure should be conducted with unbiased evaluators and hidden ranks. Such an evaluation is one of our highest priorities in further work, as the method could be valuable also in a wider context, if confirmed useful. Namely, the bi-gram ratio scoring could be applied also to other automatic slogan generation methods, and with appropriate adaptations, perhaps even in a wider array of similar problems.

Acknowledgments. This work was partly funded by the Slovene Research Agency and supported through EC funding for the project ConCreTe (grant

number 611733) and project WHIM (grant number 611560) that acknowledge the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission. The authors would also like to thank the anonymous reviewers for their valuable comments and suggestions.

References

1. Binsted, K., Bergen, B., O'Mara, D., Coulson, S., Nijholt, A., Stock, O., Strapparava, C., Ritchie, G., Manurung, R., Pain, H., et al.: Computational humor. IEEE Intelligent Systems (2), 59–69 (2006)
2. Callaway, C.B., Lester, J.C.: Narrative prose generation. Artificial Intelligence 139(2), 213–252 (2002)
3. Colton, S., Goodwin, J., Veale, T.: Full FACE poetry generation. In: Proceedings of the Third International Conference on Computational Creativity. pp. 95–102 (2012)
4. Gervás, P.: Computational modelling of poetry generation. In: Artificial Intelligence and Poetry Symposium, AISB Convention (2013)
5. Gervás, P., Díaz-Agudo, B., Peinado, F., Hervás, R.: Story plot generation based on CBR. Knowledge-Based Systems 18(4), 235–242 (2005)
6. Gervás, P., Lönneker-Rodman, B., Meister, J.C., Peinado, F.: Narrative models: Narratology meets artificial intelligence. In: International Conference on Language Resources and Evaluation. Satellite Workshop: Toward Computational Models of Literary Analysis. pp. 44–51 (2006)
7. Leake, D.B.: Case-Based Reasoning: Experiences, lessons and future directions. MIT press (1996)
8. Lenz, M.: Defining knowledge layers for textual case-based reasoning. In: Advances in Case-Based Reasoning, pp. 298–309. Springer (1998)
9. Özbal, G., Pighin, D., Strapparava, C.: BRAINSUP: Brainstorming Support for Creative Sentence Generation. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. pp. 1446–1455. Sofia, Bulgaria (2013), http://www.newdesign.aclweb.org/anthology-new/P/P13/P13-1142.pdf
10. Ritchie, G.: Current directions in computational humour. Artificial Intelligence Review 16(2), 119–135 (2001)
11. Stock, O., Strapparava, C.: The act of creating humorous acronyms. Applied Artificial Intelligence 19(2), 137–151 (2005)
12. Tomašič, P., Žnidaršič, M., Papa, G.: Implementation of a slogan generator. In: The Fifth International Conference on Computational Creativity, ICCC 2014. pp. 340 – 343. Ljubljana, Slovenia (2014), http://kt.ijs.si/publ/iccc_2014_proceedings.pdf
13. Weber, R.O., Ashley, K.D., Brüninghaus, S.: Textual case-based reasoning. Knowledge Engineering Review 20(3), 255–260 (2005)

Conceptual Blending in Case Adaptation (Position Paper)

Amílcar Cardoso and Pedro Martins

CISUC, Department of Informatics Engineering
University of Coimbra, Coimbra, Portugal

Abstract. We propose that Conceptual Blending (CB) can play a role within the Case-Based Reasoning (CBR) paradigm, particularly in the *Reuse* and *Revise* tasks of the classic model of the problem solving cycle in CBR, as an alternative adaptation mechanism that may provide suitable solutions in computational creativity setups, where novel and surprising solutions are sought. We discuss how a particular computational implementation of CB can intervene in the CBR cycle, and use the results of an experiment made in the past to illustrate the aproach. We focus our attention on graph-based structured cases. Other case representations could also be considered in the future.

1 Introduction

The Conceptual Blending (CB) theory [3] intends to explain several cognitive phenomena related to the creation of ideas and meanings. A key element in this theory is the *mental space*, which corresponds to a temporary and partial structure of knowledge built for the purpose of local understanding and action. The CB framework relies on a network comprised of at least four connected mental spaces (Figure 1). Two or more of them correspond to the *input spaces*, which are the initial domains, i.e., the content that will be blended. Then, a cross-space mapping, i.e., a partial correspondence between the input spaces, is established. The correspondences between elements of the different input spaces is not arbitrary; elements are only matched if they are perceived as similar in some way. This association is reflected in another mental space, the *generic space*, which contains elements common to the different input spaces, capturing the conceptual structure that is shared by the initial mental spaces. The result of the blending process is the *blend*, a new mental space that maintains partial structures from the input spaces, combined with an emergent structure.

In this position paper, we propose that Conceptual Blending can play a role within Case-Based Reasoning, particularly in the *Reuse* and *Revise* tasks of the classic model of the problem solving cycle in CBR, known as the "4 REs" [1], as an alternative adaptation mechanism that may provide better solutions in computational creativity setups, and possibly also for problem solving. We will focus our attention on graph-based structured cases (like in [7]), but we think the approach could also be adapted to other case representations [2]. To better

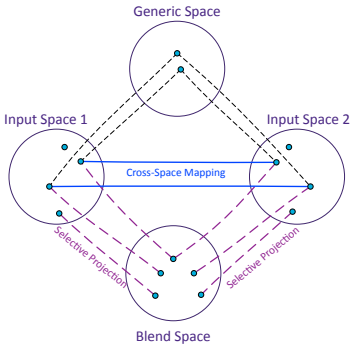


Fig. 1. The original four-space conceptual blending network [4].

explain our idea, we will use an implementation of the CB mechanism called Divago [6], previously developed by our team.

After the current introduction, we will briefly describe Divago in Section 2 and present our proposal in Section 3. In Section 4 we draw some conclusions.

2 Divago

The CB framework has served as the basis for several artificial creative systems. To discuss the role of CB within the CBR cycle, we focus on the Divago architecture [6], which relies on one of the most thorough and detailed computational models of CB to date.

The Divago framework works on a multi-domain knowledge base where the basic representation formalism is the *concept map*, a semantic network that denotes the relationship between the concepts of a given domain. It is composed of several modules (Fig. 2) that reflect the different stages of the CB mechanism.

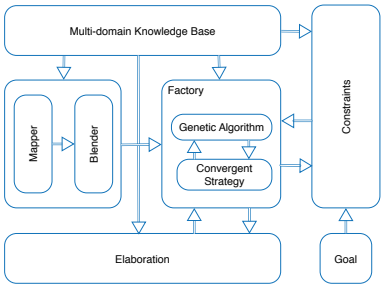


Fig. 2. Divago's architecture.

The process starts by feeding a pair of input spaces (domains) from the knowledge base into the *Mapper* module, which is responsible for performing

the selection of elements for projection. Such selection is achieved by means of a partial mapping between the input spaces using *structural alignment*. This operation looks for the largest *isomorphic* (structurally equivalent) pair of subgraphs contained in the input spaces. Each mapping is a set of mapping relations $m(x, y)$ between two concepts, one of each input space.

For each resulting mapping, the *Blender* module performs a projection operation into the blended space: for each $m(x, y)$ in the mapping, it produces a nondeterministic projection choice between x, y, $\emptyset$ and $x|y$ (which means *both x and y*); each combination of choices is the *seed* of a possible blend (to be completed and elaborated in the next stages). This process results in a graph structure (the *blendoid*) that includes all projection choices and thus represent the search space for all the blends that may result from the mapping.

The *Factory* module is responsible for exploring this search space. It is based on a variation of a genetic algorithm (GA) that uses the *Elaboration* module to enrich blends with additional knowledge and the *Constraints* module to assess their quality. This module provides an implementation of the *optimality principles* (a set of principles that ensure a coherent and highly integrated blend [3]). When an adequate solution is found or a pre-defined number of iterations is attained, the Factory stops the execution of the GA and returns the best blend. The *Constraints* module acts, thus, as the "fitness function" of the algorithm.

3 Conceptual blending in case-adaptation

The classic model of the problem solving cycle in CBR, known as the "4 REs", comprises 4 tasks: *Retrieve*, *Reuse*, *Revise* and *Retain* [1]. In the core of the process lies a *case base* of stored past experiences, each one of them comprising a problem description and the respective solution.

Although cases can be represented in many different ways [2], we will consider the situation where a structured representation is used, like for instance [7]. In particular, we will assume that there are *relations* between *attributes*. Some of them allow for hierarchical organisations (e.g., *isa* and *partwhole*), others induce a network structures (e.g., *purpose*, *shape*, relations for relative position). Table 1 describes, using a Prolog-like notation, a fragment of a case for a "House", where such relations occur. The right column is a partial description of the attribute/value pair part of the same case.

Coming back to the "4 REs" cycle, the reasoning process starts with a new problem specification being given to the first task, *Retrieve*, which seeks for stored cases with similar problem descriptions, using some similarity criterion. The result is a list of retrieved cases, of which one can be selected as having the most similar problem description to the given problem. In the general case, the similatity is not absolute and differences with the given problem description exist. This requires that the retrieved case is subject to some sort of adaptation in the task *Reuse*, trying to compensate for the differences with the given problem description. *Revise* will be responsible for evaluating the quality of the result.

Table 1. Fragment of the "House" case.

isa(house,physical_structure)	part_whole(door, house)	instance_of(r1, roof)
isa(door, physical_object)	part_whole(window, house)	instance_of(b1, body)
isa(window, physical_object)	part_whole(roof, house)	instance_of(d1, door)
isa(roof, physical_object)	part_whole(body, house)	instance_of(w1, window)
isa(body, physical_object)	part_whole(room, house)	shape(r1, triangle)
isa(observation, task)	purpose(body, container)	shape(b1, square)
isa(protection, task)	purpose(door, entrance)	shape(w1, square)
isa(entrance, task)	purpose(window, observation)	
isa(container, physical_object)	purpose(roof, protection)	

Now, let us assume that the retrieved case, c_r, is the one described in Table 1. This might happen, for instance, if the case base was composed of descriptions of houses, the problem to solve was to find a house description according to a given specification and the specification of c_r was the most similar to the given one. Let us also assume that we are in a creative setup, where we want to find ideas for houses that, although satisfying the specification, are novel and surprising. Our proposal is to seek for surprising solutions by processing the adaptation through *blending c_r with knowledge from a different domain*. The result will be a case that shares part of its description with the retrieved case, but includes contributions from the other domain. Such contributions may, for instance, fill existing gaps in c_r, substitute part of its structure, etc. As we will see, the result may be more or less *divergent* from the original domain of "houses" according to how we control the blending process and "how far" from "houses" the other domain is. The domain to use in this process may be chosen by the user, or may result from a contextual analysis whose discussion is outside the scope of this paper. We argue that Divago can deal with the process in a suitable way.

To illustrate our proposal, we re-visit the experiment described in [5], where the blend of two domains, "boats" and "houses", is explored using just the modules *Mapper* and *Blender* of Divago, with the aim of studying their *generation potential*. The situation is very similar to the one described in the previous section, as c_r, the "House" case, can be seen as an instance of the original "houses" domain. With this analysis, we intend to illustrate how the "House" case can be merged with the domain "boats".

In the experiment, the *blendoid* resulting from the most frequent mapping represents a wide variety of instances for "boat-house". We show six of them in Figure 3, where the visual representation of c_r is shown on the left.

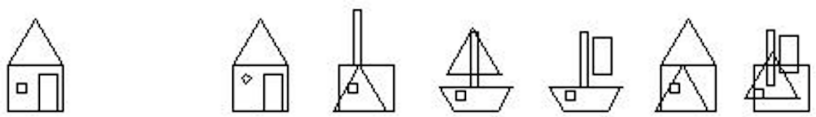


Fig. 3. The retrieved "House" case and six possible blends with the "boat" domain.

We can see that the weight of the "boats" domain in the blends varies a lot. The *divergence* of the blends from the stereotypical description of a Boat and from c_r also varies a lot, from a house with a hatch instead of a window to a house with a sail instead of a door and a mast instead of a roof.

In Divago, the GA-like search for blends is guided by an implementation of a variation of the "optimality principles" proposed in the CB theory, which favours the coherence of the resulting blends. In the context of this proposal, however, a metric for the similarity with the original problem specification should also be taken into account, and possibly assume a prevailing weight in measuring the quality of the blends.

4 Conclusions

We argued that Conceptual Blending, and in particular its computational implementation Divago, can provide an alternative adaptation mechanism for the *Reuse* and *Revise* tasks of the classic CBR model. The idea is to *blend* the case selected in the *Retrieve* task with knowledge from a different domain. This may prove especially effective in computational creativity contexts, where it may provide an iterative divergence mechanism coupled with evaluation. The criteria for evaluating each possible blend may combine measures of coherence with measures of distance to the given problem specification. This is a preliminary proposal in the context of a Position Paper. Definitely, further research is needed to understand its limits.

Acknowledgements. The authors acknowledge the financial support from the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under the ConCreTe FET-Open project (grant number 611733) and the PROSECCO FET-Proactive project (grant number 600653).

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI communications 7(1), 39–59 (1994)
2. Bergmann, R., Kolodner, J., Plaza, E.: Representation in case-based reasoning. Knowl. Eng. Rev. 20(3), 209–213 (Sep 2005)
3. Fauconnier, G., Turner, M.: Conceptual integration networks. Cognitive Science 22(2), 133–187 (1998)
4. Fauconnier, G., Turner, M.: The Way We Think. New York: Basic Books (2002)
5. Pereira, F.C., Cardoso, A.: The boat-house visual blending experience. In: Procs 2nd. Workshop on Creative Systems, ECAI 2002, Lyon, France (2002)
6. Pereira, F.C.: Creativity and AI: A Conceptual Blending approach. Ph.D. thesis, FCTUC, Universidade de Coimbra, Portugal (2005)
7. Plaza, E.: Cases as terms: A feature term approach to the structured representation of cases. In: Case-Based Reasoning Research and Development, pp. 265–276. Springer (1995)

Calibrating a Metric for Similarity of Stories against Human Judgment *

Raquel Hervás, Antonio A. Sánchez-Ruiz, Pablo Gervás, Carlos León

Dep. Ingeniería del Software e Inteligencia Artificial
Universidad Complutense de Madrid (Spain)
raquelhb@fdi.ucm.es, antsanch@fdi.ucm.es, pgervas@sip.ucm.es,
cleon@fdi.ucm.es

Abstract. The identification of similarity is crucial for reusing experience, where it provides the criterion for which elements to reuse in a given context, and for creativity, where generation of artifacts that are similar to those that already existed is not considered creative. Yet similarity is difficult to compute between complex artifacts such as stories. The present paper compares the judgment on similarity between stories explained by a human judge with a similarity metric for stories based on plan refinements. The need to identify the features that humans consider important when judging story similarity is paramount on the road to selecting appropriate metrics for the various tasks.

Keywords: similarity, novelty, stories, plans.

1 Introduction

Appropriate metrics for similarity are fundamental tools in many fields of Artificial Intelligence. For instance, there are several data mining and machine learning methods that are based on the similarity between the elements being considered. In case-based reasoning, similarity metrics are crucial for the retrieval and reuse of previous cases. Similarity is also fundamental for computational creativity because artifacts that are very similar to previously existing ones might not be considered creative. For this reason, it is important to take into account whether the metrics considered for a particular task adequately represent the concept of similarity that humans faced with the same task would apply. The present paper compares the judgment on similarity between stories explained by a human judge with a particular similarity metric for stories. The main goal is to identify which of the features that a human considers when evaluating story similarity are already taken into account by the metric, and which ones are not. The results of this comparison should provide a check list that might later on be applied to evaluate the appropriateness of other metrics.

* The research reported in this paper was partially supported by the Project WHIM 611560 funded by the European Commission, Framework Programme 7, the ICT theme, and the Future and Emerging Technologies FET programme; and by the Spanish Ministry of Economy and Competitiveness under grant TIN2014-55006-R.

We focus on the structural similarity of stories represented as plans composed of actions corresponding to the events in the story. In order to do so, we apply a similarity metric based on plan refinements and compare the obtained results for a pair of stories with the similarities found by a human expert. The key point of this comparison is that the metric does not only calculate a numerical similarity between the compared stories, but provides a report of the found similarities. This report is then compared with the observations obtained by the human expert. The comparison allows us to see if the automatic metric has been able to grasp the same features the expert considered important, and if structural similarity is enough for comparing computer-generated stories.

2 Previous Work on Similarity for Stories

Existing work on similarity for stories has focused on two different axes: story similarity for retrieval and classification of stories, and story similarity applied to the assessment of their novelty in a computational creativity setting.

2.1 Similarity Metrics for Story Generation

In general, there is relative consensus on the fact that comparing stories can be made at different levels. Comparing stories at a relatively abstract level is common, to the point of comparing not the exact sequence of events but the overall plot, or even the relations between the characters. This aspect of narrative has been addressed by structuralist and cognitive Narratology.

In particular, comparing narratives has been a long term goal of Computational Narrative, and several approaches have been taken with varying results [2, 10, 8]. Different aspects beyond pure literary composition have been tackled: structure alignment in bioinformatics [1], event mapping [3], and other approaches like considering story similarity in terms of the common summary that might be abstracted from the two stories being compared [9].

2.2 Similarity Metrics for Assessing Novelty of Stories

With respect to the assessment of creativity, a fundamental pillar is whether the results of a creative process have produced novel artifacts [14]. Research on the evaluation of creativity has addressed this point as an important requirement for the scientific exploration of creativity, and an important one for computational approaches. In [11], novelty of a given story is assessed in terms of new elements that appear in the story, or instances where existing elements have been replaced by elements of a different type. In [12], novelty of stories is considered in terms of their differences with an initial set of reference stories, based on the sequence of actions, the structure of the story in terms of emotional relations and tensions between the characters, and the occurrence of repetitive patterns.

Story 1	Story 2	Common structure
shows id371 id372	**declare-war id818 id819**	declare-war id818 id819
offers-exchange id371 id372 id373	sings id207 murder	decides-to-react ?x1
not-perform-service id373	**decides-to-react id142**	sets-out ?x1
negative-result id373	**sets-out id142**	wins ?x1
consumes id373 id44	**wins id142**	brings-peace ?x1
acquires id373 magical-abilities	**brings-peace id142**	arrives ?x1 ?x2
declare-war id818 id819	**arrives id142 id730**	disguised ?x1
dispatches id189 id373	**disguised id142**	unrecognised ?x1
tells id189 id373 past-misfortune	**unrecognised id142**	claims id672 won id818
decides-to-react id373	**claims id672 won id818**	sets ?x3 ?x1
sets-out id373	**sets id165 id142**	involves difficult-task ?x4
wins id373	**involves difficult-task strength**	solve ?x1 difficult-task
brings-peace id373	**solve id142 difficult-task**	before dead-line
arrives id373 id728	**before dead-line**	returns ?x1
disguised id373	**returns id142**	arrives ?x1 id730
unrecognised id373	**arrives id142 id730**	disguised ?x1
claims id672 won id818	**disguised id142**	unrecognised ?x1
sets id161 id373	**unrecognised id142**	claims id672 won id818
involves difficult-task kissing	**claims id672 won id818**	exposed id672
marked id373	**exposed id672**	not-solve id672 difficult-task
solve id373 difficult-task	**not-solve id672 difficult-task**	
before dead-line	new-physical-appearance id142	
returns id373	punished id818	
arrives id373 id730	tied-to id818 horse-tail	
disguised id373		
unrecognised id373		
claims id672 won id818		
exposed id672		
not-solve id672 difficult-task		

Table 1: Table of events in each of the stories and the shared set of events.

3 A Callibration Exercise for Story Similarity

Although there are many possible representations for stories and many different metrics have been considered for story similarity, the present effort has been focused on a particular representation format as used by an existing story generator, and a specific metric that allows automatic computation. These choices were circumstantial on ease of access and are not considered optimal, but the effort should produce valuable insights that can later be extended to other alternatives.

3.1 Story Representation in the Propper System

The Propper system [5] constitutes a computational implementation of a story generator based on Propp's description of how his morphology might be used to generate stories [13]. It produces stories as a sequence of states described in terms of predicates that hold in the state. Characters, objects or locations are represented as unique identifiers in the predicates. This representation format has been considered generic enough to allow for an initial calibration exercise, considering that other formats may easily be converted into this one.

The representation includes predicates representing narrative events and predicates describing properties of the characters that hold in particular states

of the story. These appear jointly in the stream of predicates for the story, but have been separated in the presentation of stories in this paper for clarity.

The predicates presented here result from an effort of reverse engineering of the stories that Propp describes as examples of the application of his framework to analyse existing Russian folk tales.

The first two columns of Table 1 present two examples of the stories produced by the Propper system. Predicates in this table describe actions or events in the story. Table 2 represents non-narrative facts that are true for the arguments of the actions in Table 1.

Story 1	Story 2	Common structure
hero id373	*villain id818*	*villain id818*
donor id371	*victim id819*	*victim id819*
magical-agent id372	*hero id142*	*hero ?x1*
magical-agent id44	*seeker-hero id142*	*location ?x2*
villain id818	*location id730*	*false-hero id672*
victim id819	*court id730*	*unknown ?x3*
seeker-hero id373	*groom id142*	*task-type ?x4*
dispatcher id189	*false-hero id672*	*court id730*
location id728	*location id730*	
home id728	*court id730*	
apprentice id373 artisan	*groom id142*	
false-hero id672		
location id730		
court id730		
groom id373		

Table 2: Table of characters, locations and objects in the two stories and the shared set

3.2 Human Interpretation of the Stories

In order to compare the human interpretation of the stories with an automatically extracted report, we asked a human expert to write both stories in English and compare them. It is important to mention that the expert was familiar with this type of representation based on predicates, but she had to figure out the meaning of the predicates based solely on their names.

Story 1

This story has the following main characters: a hero (373), a villain (818), and a false hero (672). In addition, a donor (371), a victim (819) and a dispatcher (189) appear as secondary characters.

The hero (373) is first offered a magical agent by a donor (371) if he performs a service. He does not perform the service but he obtains another magical agent anyway, which he consumes to acquire magical abilities.

Then, a villain (818) appears who declares war to a victim (819). The victim does not appear again.

Meanwhile, a dispatcher (189) talks about a past misfortune. The hero decides to react, sets out and wins (the war?), bringing peace with him. After that, the hero goes home, but he is disguised as the apprentice of an artisan and is not recognised. He finds a false hero (672) at home, who claims that he defeated the villain.

The hero is marked, solves a difficult task and returns to the court, this time disguised but as a groom. The false hero still claims that he defeated the villain, but he is exposed and it is known that he did not solved a difficult task.

Story 2

This story has the following main characters: a hero (142), a villain (818), and a false hero (672). In addition, a victim (819) appears only at the beginning.

The story starts with the villain (818) declaring war to the victim (819). The hero (142) decides to react, becomes a seeker hero, sets out and wins (the war?). He brings peace and arrives to the court. But he is disguised as a groom and he is not recognized.

At the court, the false hero (672) claims that he defeated the villain. Someone (165) sets the hero a difficult task that involves strength. He solves the difficult task before the deadline, and returns to the court. Again he is disguised as a groom and he is not recognized.

And again, the false hero claims that he defeated the villain. However, the false hero is exposed and does not solve a difficult task. The hero gets a new physical appearance (undisguised?), and the villain is punished being tied to a horse tail.

Next, we asked the expert to compare both stories and describe the main similarities and differences between them.

Both stories are similar in their characters and roles: a hero, a villain, and a false hero who claims to have defeated the villain.

In addition, in both stories the villain declares war to a victim, and the hero wins the war and brings peace. After that the hero returns (home or to the court) disguised (as a groom or as an apprentice), and he finds that a false hero claims to have defeated the villain. But at the end the false hero is exposed in both stories. Also, in both stories the hero makes two different journeys: one to win the war and return home/court, and one to solve a difficult task and then returning to court.

From the point of view of the differences, Story 1 involves magic. The hero tries twice to obtain a magical agent, and the second time he achieves it and gets magical abilities. However, they are not used in the story. The main difference in Story 2 is that at the end the villain is explicitly punished by being tied to a horse tail.

It is interesting to note that the first things mentioned by the expert both in the descriptions and the comparison are the characters, although in the comparison only the most important characters are mentioned, as the others are considered less important for the plot.

In addition, the descriptions are based on the most important events in the story, so not all events are considered equally important. The comparison also shows that there is a high similarity between both stories in terms of characters and some of the narrative arcs. For example, the hero returns in both stories but to different places and with different disguises. However, these differences (place and disguise) are not considered as important and the expert finds similarity in what is happening even when the stories are not exactly the same.

One of the main differences between the stories is that one of them involves magic, but it is not considered so important because magic is not used in the rest of the story. Finally, the differences in the endings are explicitly addressed in the comparison. This means that the end of the story is an important part of it.

3.3 Computing the Common Structure of Two Stories using Plan Refinements

A story in its more basic form can be represented as a sequence of actions, i.e., as a *plan*. There are different approaches to compute the similarity of two plans. In this paper we use the similarity measure based on plan refinements presented in [15] because it does not only provide a numerical similarity value but an explicit description of the common structure shared by both plans. This common structure can be seen as a directed graph in which each node represents an action and each directed edge represents an ordering constraint. Two actions are connected in the graph only if both actions appear in that order in the plans being compared.

Besides the actions and their order, this similarity measure also considers the action parameters and, if they are different in both plans, it is able to infer their common type according to a domain taxonomy. In this way, we are able to detect objects, characters and locations in different stories that have a different name but play the same role in the story.

The similarity measure computes this common structure performing successive refinements in the space of partial plans [7]. There are five different types of refinements that specialize a partial plan: to add a new action, to add a new ordering constraint between two existing actions, to specialize the type of a variable representing an action parameter according to a domain taxonomy, to unify two different variables, and to replace a variable with a domain constant.

The similarity measure works as follows. Let us suppose we want to compare two plans (or stories) p_1 and p_2. The similarity measure begins with an empty partial plan (a plan with no actions) that represents any possible plan and thus it is more general than p_1 and p_2. Then the partial plan is specialized using a refinement operator (adding new actions and ordering constraints or specializing the action's parameters) until we reach another partial plan that cannot be specialized anymore while being more general that both p_1 and p_2. This partial plan is the *most specific generalizer* of p_1 and p_2, $MSG(p_1, p_2)$, and represents the common structure shared by the two plans. The length of the refinement chain from the empty plan to the $MSG(p_1, p_2)$ is an indicator of how similar

the two plans are. In the same way, the length of the refinement chain from the $MSG(p_1, p_2)$ to each one of the two plans is an indicator of how much information is contained only in one of them but not in the other. The similarity value is computed as the ration between the amount of information shared and the total amount of information contained in the two plans.

The last columns of Tables 1 and 2 show the common structure computed by the similarity measure. In this case, the two stories are very similar and the inferred common graph of actions is so simple that, in fact, it can be represented as a sequence of actions. Constants representing characters, locations and objects common to both stories are kept in the common structure, and the other constants are replaced by variables with generalized types (variable names begin with '?').

The common structure of both stories could be summarized as follows. A villain declares war on a victim, what triggers the intervention of a hero that defeats him and brings peace back. Then the hero travels disguised and see how a false hero claims that he, and not the original hero, has defeated the villain. The hero leaves, solves a difficult task before some deadline, and comes back disguised. The false hero is exposed in court because he was not able to solve the difficult task.

4 Discussion

There are a number of issues that the similarity metric considered here does not take into account.

First, the point in the story in which a particular sequence of actions takes place may lead to different results. A marriage at the start of the story sets the scene for later actions, but at the end of the story it usually acts as a reward for the efforts of some character. This influence of context is not considered in the metric that has been described.

Second, some events are more significant than others. The presence of a murder in a given story is more significant than that of more mundane events such as setting off on a journey. This aspect might be captured by some kind of weighting of the importance of specific events. The described metric does not allow for this type of behaviour.

The judgment expressed by the human placed considerable emphasis on the relative importance of the elements that appear in the stories. Characters are mentioned first, then specific actions. In both cases, a certain degree of abstraction is applied to identify conceptual similarity even between instances that are different. This suggests that taxonomical reasoning might be a useful tool for assessing similarity and that, as expected, abstraction is fundamental in story similarity.

These two aspects suggest that automatic story comparison needs to address *lifting* between different levels of abstraction to be able to match those features that humans are able to match. It also seems that the abstract matching at different levels is a fundamental cognitive tool for comparing stories in humans.

This conclusion relates to the approach in [9] of considering similarity between stories in terms of a shared summary, but extended to summarisation with an important degree of abstraction. The work in [11], by virtue of being based on description logic ontologies, does include the possibility of taxonomical reasoning being applied in the process of measuring similarity. It is clear that this particular approach should be explored in more detail in future work.

The version of the Propper system that has been employed here provides only limited description of the characters. The descriptions considered are restricted to specification of the roles played in the narrative by particular characters, and a number of properties of particular arguments that are relevant for the correct chaining of later actions with their context of occurrence via their set of preconditions.

An important problem from the point of view of assessing the novelty of creative processes is the need to consider an existing set of artifacts as a reference. Generated artifacts are only novel if they are not similar to existing ones. However, from a computational point of view, the approach of keeping a record of all existing artifacts of a given type, and computing the similarity of any newly generated artifacts with this set is not practical [4]. Indexing solutions may be used to improve efficiency, but even so, solutions based on some level of abstraction, away from specific instances and addressing more generic characterisations of the artifacts (in this particular case, stories) would prove more practical in this context. Conformance or departure from Concepts such as conventional endings, genre conventions, or character stereotypes may play a fundamental role in assessing the novelty of stories beyond sequences of actions.

Overall, it seems that there are a number of aspects of stories that are relevant when attempting to establish similarity between two instances of story. Just how many such aspects should be included in a particular implementation as a similarity metric may depend substantially on the purpose for which it is intended. In the particular case of similarity metrics employed for case-based reasoning, the choice of which aspects of similarity to model should be guided by the particular aspects of the case that will be reused. If the cases are intended to provide story structure, the similarity should focus on story structure. If the cases are intended to inform decisions on the set of characters to employ, the similarity should focus on the set of characters. In relation to the point raised above concerning abstraction, it is important to note that focusing on particular aspects of story similarity may require specific types of abstraction to implement the described lifting operation. Where similarity metrics are used for evaluating novelty in Computational Creativity settings, their use is much broader and it becomes more difficult to focus on particular aspects. Nevetheless, as it is very important to consider issues of efficiency, abstraction as means of reducing the range of attributes that need to be compared will clearly play a fundamental role in practical implementations.

5 Conclusions and Future Work

The present work describes a process by which a computational system for computing the similarity between narrative structures is compared and calibrated against human judgment.

A number of issues considered by the human judge but not covered by the system have been discovered. These should be considered as a check list for the consideration of alternative metrics, and possibly as driving guidelines for the development of more elaborate metrics specific to the assessment of story similarity.

The work described in this paper has addressed sequential single narrative threads. More complex narratives usually involve parallel story lines which merge or split at several points in the overall narrative. Whether the current metrics are valid for comparing similarity between this kind of narratives or not is yet an open question. Additionally, the use of different structures for stories also opens a new path, namely the application of the current process to stories that, while outputting an equivalent format, are generated by other story generation systems, probably conveying different semantics in the sequence of events, and possibly richer relations between characters.

From this point of view, more recent versions of the Propper system [6] address specifically the description of characters as they occur in the story, and they should be explored in further work to extend the metric for similarity to consider differences between the characters of two stories. For that work, it may be necessary to focus on differences between characters fulfilling equivalent narrative roles in the different stories.

State is also fundamental in narrative composition and analysis. Narrative understanding of statements like "John squashed the spider" heavily depend on the relation between John and the spider (was it his mascot?). This kind of information must be taken into account in a general model of story similarity.

In all cases, further research must look into more metrics for story comparison and employ more experts to analyse how humans evaluate narratives. Following the intuition that we, as humans, perform a complex set of comparisons for evaluating similarity at different levels can lead to the discovery of plausible metrics and plausible aggregation methods into one single judgment.

References

1. Fay, M.: Story comparison via simultaneous matching and alignment. In: Workshop on Computational Models of Narrative, 2012 Language Resources and Evaluation Conference (LREC'2012). Istambul, Turkey (2012)
2. Fisseni, B., Lowe, B.: Which dimensions of narratives are relevant for human judgments of story equivalence? In: Workshop on Computational Models of Narrative, 2012 Language Resources and Evaluation Conference (LREC'2012). Istambul, Turkey (2012)
3. Fisseni, B., Lowe, B.: Event mappings for comparing formal frameworks of narratives. Logique et Analyse (57) (2014)

4. Gervás, P.: Dynamic inspiring sets for sustained novelty in poetry generation. In: Second International Conference on Computational Creativity. México City, México (2011)
5. Gervás, P.: Propp's morphology of the folk tale as a grammar for generation. In: Workshop on Computational Models of Narrative, a satellite workshop of CogSci 2013: The 35th meeting of the Cognitive Science Society. Universität Hamburg Hamburg, Germany (2013)
6. Gervás, P.: Computational drafting of plot structures for russian folk tales. Cognitive Computation (2015)
7. Kambhampati, S., Knoblock, C.A., Yang, Q.: Planning as refinement search: A unified framework for evaluating design tradeoffs in partial-order planning. Artificial Intelligence 76(1), 167–238 (1995)
8. Krakauer, C., Winston, P.: Story retrieval and comparison using concept patterns. In: Workshop on Computational Models of Narrative, 2012 Language Resources and Evaluation Conference (LREC'2012). Istambul, Turkey (2012)
9. Kypridemou, E., Michael, L.: Narrative similarity as common summary: Evaluation of behavioral and computational aspects. LLC 29(4), 532–560 (2014), http://dx.doi.org/10.1093/llc/fqu046
10. Michael, L.: Similarity of narratives. In: Workshop on Computational Models of Narrative, 2012 Language Resources and Evaluation Conference (LREC'2012). Istambul, Turkey (2012)
11. Peinado, F., Francisco, V., Hervás, R., Gervás, P.: Assessing the novelty of computer-generated narratives using empirical metrics. MINDS AND MACHINES 20(4), 588 (2010)
12. Pérez y Pérez, R., Ortiz, O., Luna, W.A., Negrete, S., Pealoza, E., Castellanos, V., vila, R.: A system for evaluating novelty in computer generated narratives. In: Proceedings of the Second International Conference on Computational Creativity. pp. 63–68. Mxico City, Mxico (2011)
13. Propp, V.: Morphology of the Folk Tale. Akademija, Leningrad (1928)
14. Ritchie, G.: Some Empirical Criteria for Attributing Creativity to a Computer Program. Minds & Machines 17, 67–99 (2007)
15. Sánchez-Ruiz, A.A., Ontañón, S.: Least common subsumer trees for plan retrieval. In: Case-Based Reasoning Research and Development - 22nd International Conference, ICCBR 2014, Cork, Ireland, September 29, 2014 - October 1, 2014. Proceedings. pp. 405–419 (2014)

Creative Systems as Dynamical Systems

Alessandro Valitutti

School of Computer Science and Informatics, University College Dublin,
Belfield, Dublin D4, Ireland
alessandro.valitutti@ucd.ie

Abstract. In this paper, we discuss ideas for characterizing a case-based generative system as "creative". Focusing on a specific generator of graphics, we performed a qualitative exploration of the space of solutions. The emerged intuition is that the set of configurations generated by the program can be viewed both as the conceptual space of a creative system and the phase space of a dynamical system. In the context of this analogy, we hypothesize that a higher degree of creativity can be ascribed to the search paths allowing the system to reach new basins of attractions.

1 Introduction

Case-based reasoning (CBR) is a type of problem solving in which a new solution is found through the retrieval of a similar available case and the adaptation of the related solution [1].

Let us suppose to have a computer program for the generation of artworks such as graphics, musical pieces, or poems, and a set of generative parameters. Given a set of known examples, a different initialization of the parameters should allow the system to produce different corresponding instances of the same type of artifact. However, the production of new artifacts does not necessarily imply that they would be recognized as original and valuable. In this paper, we discuss ideas for characterizing the re-use of past solutions, performed by a case-based generative system, as "creative".

An artwork generator can be framed in the context of ideas on *creative systems* introduced by Boden [2], formalized by Wiggins [12] and further extended by Ritchie [11]. In this context, the case-based adaptive process can be viewed as a type of *exploratory creativity*, i.e. a search in the space of artifacts or *conceptual space*, where the set of past examples are the *inspiring set*. Ideally, the output of the search should be an artifact provided with a form of value and expressing the balance between familiarity and novelty described by Giora as *optimal innovation* [4].

Focusing on a specific generator of graphics, we performed a qualitative exploration of its generative parameters, described in the next section. The rest of the paper discusses the insights inspired by this example.

2 Exploring the Space of Fractal Trees

We focused on an algorithm for the visual representation of a *fractal tree*, a fractal geometrical shape defined by recursion as follows: *(1) Draw a trunk; (2) At the end of the trunk, split by some angle and draw a prefixed number of branches; (3) Repeat at the end of each branch until a sufficient level of branching is reached*[1]. The original code of the program[2] was implemented in *Processing* programming language [10]. For the mathematical details, we refer the reader to Mandelbrot's treatment [8, pp.151-161]. The shape depends on the value of two parameters representing the angle between two adjacent branches and the rotation angle performed on both of them, respectively. Their values are associated to the two coordinates of the mouse cursor in the output window. In this way, moving the cursor in different points of the screen, it is possible to generate an unlimited number of configurations.

In order to show the set of possible configurations in a small portion of the output window, we modified the code in such a way to draw a small square and to map the configurations to the coordinates of its internal points.

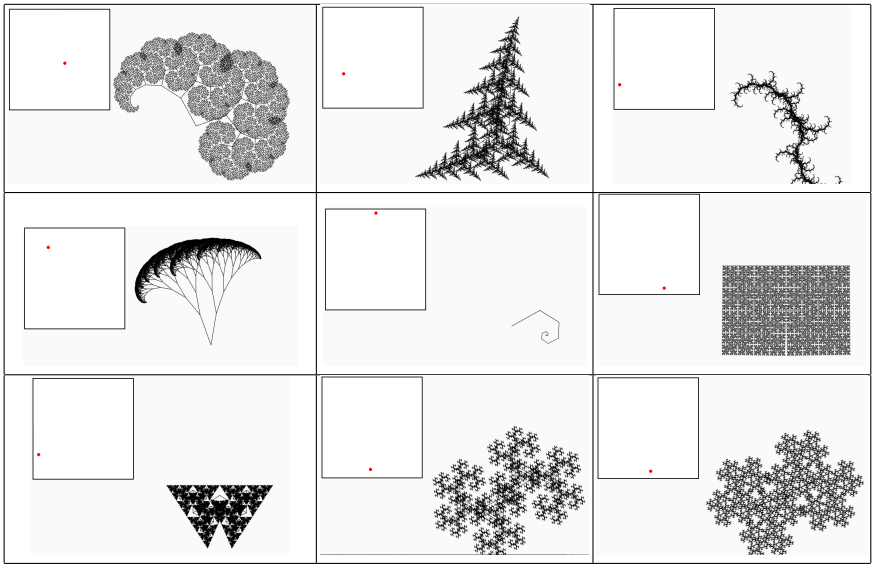


Fig. 1. Examples of configurations generated by the position of the cursor in different regions of the conceptual space mapped in the square.

[1] This version of the algorithm description is reported on `http://rosettacode.org/wiki/Fractal_tree`

[2] The code of the original program is available at `http://www.openprocessing.org/sketch/5631`.

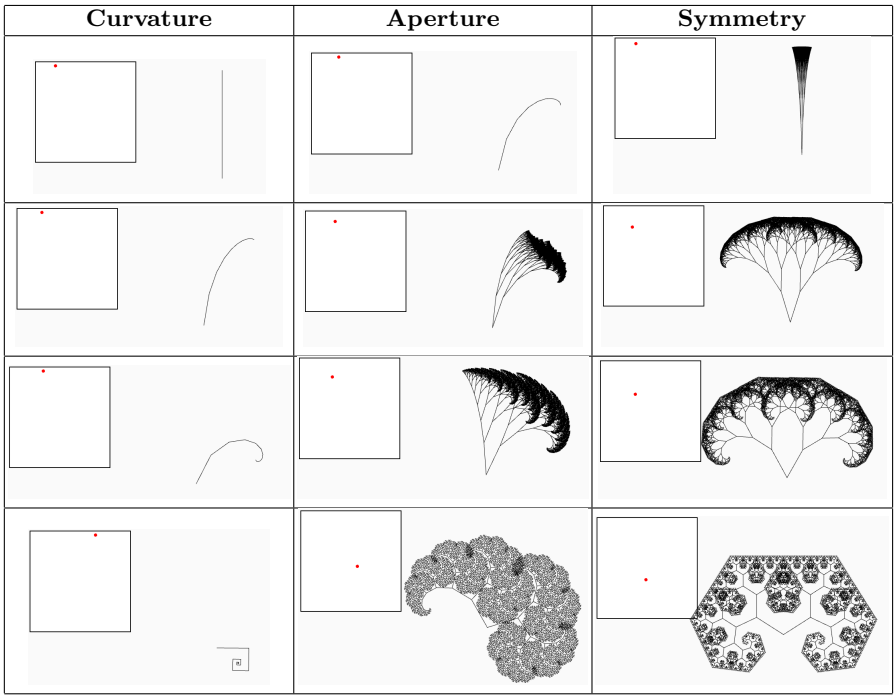


Fig. 2. Configurations according to different dimensions.

We observed the changes of the shape while moving the mouse cursor over the square. In doing that, we were inspired from a qualitative exploration described by Douglas Hofstadter in what he called an "exotic trip". He put his description in "*Gödel, Escher, Bach*" [5, pp.483-488] as a fictional dialogue and, three decades later, as a more detailed report [6, pp.65-69]. Hofstadter used a video camera pointed in various ways toward the output screen, and capable of generating several possible patterns. In particular, we made three main observations.

Shape Types Our first finding was that there are regions in the square corresponding to different types of shapes. As shown in Figure 1, some regions generate shapes recognizable as vegetable forms such as stone pines, firs, broccoli, or roots. Other regions generate polygons such as triangles, rectangles, or polygon spirals. Finally, there are regions associated to more complex shapes resembling snowflakes. Each region seems to correspond to specific *"natural concept"*, as defined by Gärdenfors [3].

Shape Dimensions The second observation is that, in each region, the shapes can be associated to a number of perceptual dimensions ascribable to Gärdenfors' *"quality dimensions"*. Specifically, we identified three dimensions: *curvature*, *aperture*, and *symmetry*. Each dimension seems to identify a specific

trajectory in the conceptual space. Figure 2 shows some configurations according to the observed dimensions. Curvature and aperture can be easily defined in terms of the generative parameters. For example, since the overall figure is the superposition of a fixed number of broken lines, curvature can be defined as the angle formed by two adjacent segments in the broken line. According to the first column of Figure 2, the trajectory of curvature is a horizontal line. Moreover, aperture can be defined as the average difference between the curvature of two adjacent components. In the case of symmetry, the definition in terms of generative parameters seems more naturally definable "a posteriori", as a constraint on the generated shape.

Optimal Configurations Finally, the third observation is that, in each region associated to specific type of shapes, the aesthetic value of the shapes seems to change according to different generative parameters and dimensions. Furthermore, each column of Figure 2 shows that the aesthetic value seems to reach a maximum in correspondence of specific subsets of each region. These "optimal configurations" seem to be associated to specific ranges of curvature, aperture, and symmetry. At this stage of the research, this claim is proposed as an intuition to be formalized and empirically evaluated. In particular, it would be necessary to attempt a formal definition of aesthetic value in terms of the shape dimensions mentioned above. Moreover, an evaluation with human judges is needed to study to what extent there is agreement on the aesthetic values and their variation along the different shapes. Specifically, we intend to employ type of evaluation with subjects analogous to the one performed by Noy et al. [7]

3 Basin Jumping

If we consider a specific path in the square mapping the conceptual space, such that the variation of the aesthetic value is positive and reach its maximum in correspondence of the optimal configurations, we can view it from two different perspectives. On one hand, the path can describe a search session in the conceptual space of a creative system. On the other hand, it can be interpreted as a trajectory in the phase space of a dynamical system. According to the second interpretation, we can view each region of the conceptual space, associated to different shape types, as *basins of attraction* and their optimal configurations as the corresponding *attractors*. An attractor is a set of states (i.e., elements of the state space of a dynamical system) towards which a set of dynamical paths tend to evolve [9]. We go beyond the specific example described above and suppose that there is a large number of creative systems whose conceptual spaces can be decomposed in basins of attraction. Moreover, we hypothesize that the "creativity" of these system should not simply consist of the capability to generate the conceptual space and, starting from an initial configuration, explore its basin of attraction. Indeed, they should be capable of reaching basins of attraction not containing the past examples. In other words, if we assume the creativity as a search in the conceptual space, a higher degree of creativity is associated to the search of new basins of attraction.

4 Learning to Jump

The intuitions proposed in this work are aimed to identify a possible limitation in the use of CBR as a creative tool and to overcome it. A creative CBR system should get a the description of an artifact (i.e. an element of the conceptual space) as input case and retrieve one or more similar cases and reuse the corresponding knowledge to generate them. A possible intrinsic limitation is the use of similarity of past solutions. In terms of dynamical systems, we believe that this approach constraints the search inside a single basin of attraction. The suggestion emerged from the example described above is to identify perceptual dimensions and, through them, evaluation functions capable of reaching the maximum value in different basins of attractions.

In our next work, we aim to formalize, implement and empirically evaluate this approach. In particular, we intend to focus on generative systems analogous to the fractal tree generator and provide definitions of perceptual dimensions and aesthetic value. A crucial aspect is the combination of two types of heuristics, the first one for the discovery of new basin of attraction, and the second one for the identification of the optimal configuration.

Acknowledgments This research was supported by the EC project *WHIM: The What-If Machine*. See http://www.whim-project.eu.

References

1. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications 7(1), 39–59 (1994)
2. Boden, M.A.: The Creative Mind. Abacus, London (1990)
3. Gärdenfors, P.: Conceptual spaces as a framework for knowledge representation. Mind and Matter 2(2), 9–27 (2004)
4. Giora, R.: On Our Mind: Salience, Context and Figurative Language. Oxford University Press, New York (2003)
5. Hofstadter, D.R.: Gödel, Escher, Bach: an Eternal Golden Braid. Basic Books, New York (1979)
6. Hofstadter, D.R.: I am a Strange Loop. Basic Books (2007)
7. Lior Noy, Y.H., Andrew, N., Ramote, O., Mayo, A., Alon, U.: Computers can't jump? A quantitative approach for studying creative leaps. In: International Conference on Computational Creativity (ICCC). pp. 72–76 (2012)
8. Mandelbrot, B.: The Fractal Geometry of Nature. W. H. Freeman, San Francisco (1982)
9. Milnor, J.W.: Attractor. Scholarpedia 1(11), 1815 (2006)
10. Reas, C., Fry, B.: Processing: A Programming Handbook for Visual Designers and Artists for Visual Designers and Artists. MIT Press (2007)
11. Ritchie, G.: Some empirical criteria for attributing creativity to a computer program. Minds and Machines 17(1), 76–99 (2007)
12. Wiggins, G.A.: Searching for computational creativity. New Generation Computing 24(3), 209–222 (2006)

Schematic processing as a framework for learning and creativity in CBR and CC

Kat Agres and Geraint A. Wiggins

Queen Mary University of London, London E1 4FZ, United Kingdom,
{kathleen.agres,geraint.wiggins}@qmul.ac.uk

Abstract. There is a clear connection to be made between psychological findings regarding learning and memory and the areas of case-based reasoning (CBR) and computational creativity (CC). This paper aims to encourage researchers in these areas to consider psychological perspectives while developing the technical and theoretical aspects of their computational systems. To this end, an overview of knowledge structures and schematic processing is provided, offering findings from music cognition to demonstrate the utility of this approach. Examples of musical schemata are offered as cases which may be used in CBR systems for combinatorial creativity and the generation of new creative output.

Keywords: cognitive psychology, schematic processing, computational creativity, case-based reasoning

1 Introduction

Creativity relies heavily upon domain-relevant experience and knowledge: an expert chess player's creative problem-solving, for example, is based on his robust knowledge and flexible thinking within his domain. Given the prime importance of past learning and experience for future creative behavior, there is an obvious marriage between the areas of case-based reasoning (CBR) and computational creativity (CC). While this connection has been explored in various computational settings, few approaches import findings and perspectives from cognitive psychology (although, see [10]), a field which may offer rich insight into this endeavour. Specifically, the mechanisms underlying learning and memory, and the way in which information is represented in the mind, should be considered, as these can elucidate creative behavior and inspire new ways of approaching machine creativity. In other words, artificial systems simulating human learning and memory can form the foundation for CBR approaches to CC.

This paper takes the stance that considering psychological mechanisms is essential not only for understanding human creativity, but for a theoretical understanding of creativity that can inform the implementation of creative processes in artificial systems. That is, researchers may be able to bolster CC by understanding how humans are creative. We focus on schematic processing mechanisms, such as the encoding and updating of memory representations, and the domain of music is considered as an example of how the abstraction of instances or cases yield schemata (e.g., generalized cases) which may be applied to CC.

2 Knowledge structures in human cognition

Cognitive psychology has thoroughly investigated learning and memory. Researchers once believed memory to be vast and detailed [28], but recent findings highlight its incompleteness and malleability. For example, vision research suggests that viewers primarily encode the general schematic attributes of a visual scene upon brief initial viewing [20, 26], supplying a semantic understanding of the scene [19, 20] but lacking detail. Similary, psychology and cognitive science have recently emphasized the importance of association and analogical processing [1,11]. Although veridical representations are sometimes encoded, more often we form general or associative semantic representations (*schemata*) of new input based on prior experience. This *schematic processing* is based on abstracted mental representations that structure or organize some aspect of past experience, and schematic memory structures influence the processing of new information.

Investigations of schematic processing have contributed to our theories of learning and memory for nearly a century [2, 24]. In *Remembering*, Bartlett notes that when individuals are asked to recall an odd or supernatural story after a time delay, their recollections alter the story to better conform to their existing schematic knowledge [2]. In other words, our knowledge shapes our perception and interpretation of the world. Piaget, who considered schemata to be the building blocks of knowledge, discussed how new information is incorporated into existing schemata in the processes of *assimilation* [24]. When the new information is too dissimilar to be integrated, *accommodation* occurs, in which the schematic structure itself must change to accommodate the new information.

The notion of schemata has been echoed in the fields of computer science and artificial intelligence for decades, for example, in Minsky's *frames* [17], and Schank's *script-based systems* [27]. Recent computational models learn and generalize the statistics of a training corpus (building what is essentially a statistical version of a schematic framework) in order to evaluate or categorize new instances [13,23]. This is akin to the process of assimilating new information into schematic representations, where the schemata in this instance are encoded in the network of probabilities underlying common structures or patterns. These statistical models have been used to generate new, creative output [22, 25]. CBR and CC approaches have successfully used techniques such as inductive analogical processes [21] and template-based methods (e.g., Gervas' ASPERA system [8]) for creative generation, but the connection to schema theory is often only implicit. Arguably, psychological findings should be explicitly applied here, because knowledge of how mental representations are formed and change over time (and are re-represented) can inform how AI systems may represent the information and knowledge required to achieve creative behaviors.

3 Music as an example domain

To show how psychology can inform how systems learn, represent, and combine information in new ways, we consider the domain of music. In the auditory modality, Bregman, Dowling, Cuddy, and others have explored the contribution of schema-based mechanisms to the abstraction of tonal relationships dur-

ing music perception [4, 5]. Experience listening to common musical patterns or forms creates our mental framework for processing music [9, 16]. The underlying schemata are essentially collections of rules that guide listeners' perception of music (and thence the information encoded) by directing attention and continually creating expectations about the forthcoming music [12, 15, 18, 23]. Although musicians may have more elaborated schemata than non-musicians, everyone exposed to music has implicitly learned musical schemata. Conversely, every schema is modified by perceptual experience, as new information is abstracted and integrated into long-term schematic memory [29].

For concrete examples of musical schemata, we may consider Gjerdingen's examples of musical schemata: the "gap-fill" schema and the "changing-note" schema [9]. The former matches a melodic leap followed by an ascending or descending sequence of tones that fills the gap created by the interval leap. The latter matches two pairs of notes, in which the first pair leads away from the tonic pitch, and the second leads back. Even musically untrained listeners are capable of distilling these schemata from examples containing both types [9]. He further argues that musical schemata comprise a specific set of features that create a *style structure* [18]. Similarly, Snyder [29] describes musical schemata as networks of long-term memory associations that are amalgamations of the statistical properties of music: semantic frameworks constructed from "the commonalities shared by different experiences" [29]. Over time, episodic memories gradually form a generalized schematic representation in which specific details of each instance are lost, but generalizability of the schemata is gained.

In sum, musical schemata are mental frameworks of musical knowledge that are abstracted from experience and guide musical expectation. One insight from this work for CBR is to not simply match cases, but to *generalise* cases into schemata. If a CBR system has internalized schemata based on a corpus of musical cases (e.g., melodies), it is equipped to process new examples with more sophistication: by extracting schematic representations of these melodies, the representations may be more easily compared, and the generation of new music is made more feasible. Consider a system that generates novel, high-quality harmonization. First, it is provided with a case base of well-harmonized melodies from which it extracts schemata and derives characteristics of good harmonization. Then, given a new melody (case), it can generate harmony by matching within the space of schemata, to extrapolate a novel but appropriate harmony.

4 Knowledge structures as the foundation for creativity

Learning mechanisms and knowledge representations (such as schemata) are essential to how humans structure and combine information. They are also of central importance to CC, and the principle of combining existing knowledge into novel ideas has been a cornerstone of creativity research for decades [3, 6, 14]. Koestler describes creativity as *bisociation*—"interlocking of two previously unrelated skills, or matrices of thought" [14]. Inspired by Koestler, Fauconnier and Turner [6] offer a cognitive theory of conceptual blending, in which elements and relationships from different sources are combined to produce new meaning.

Several authors also refer to conceptual spaces which may be combined, manipulated, and traversed [3,7,30]. In all of these approaches, schemata could be used as general cases (or matrices or regions of conceptual spaces) that may be combined to form new, creative ideas. Further, schemata may be viewed as methods for caching or even hashing the case base, thus improving retrieval efficiency.

Knowledge of psychological processes can inform how learning and memory may be instantiated in artificial systems, which in turn influences how concepts may be blended and combined. One may consider schemata to be the building blocks for exploratory and combinatorial creativity. If a CBR system maps melodic onto schematic representations, the system may then be used to classify or even generate new examples through extrapolation (or interpolation) of existing cases. This approach is especially useful for CC, because a means of reflection or self-evaluation should be built into the system, and CBR can satisfy this need. Further, the way in which humans learn and encode information can suggest particular schemata that may contribute to CC in AI systems, but also (and just as importantly), elucidate the *processes* underlying the combination of knowledge structures [30]. For example, one could use a schema-based system to judge whether new melodies will sound novel to listeners by examining whether different melodies abstract to the same schemata, and this could be very useful for applications such as automatic composition.

5 Conclusion

We argued for the consideration and inclusion of psychological findings in CBR as a means of approaching CC. Using examples of mental knowledge structures and schematic processing mechanisms in the musical domain, we discussed how existing schemata may be considered as cases for the combination of ideas and generation of new creative output. Understanding how humans learn and form memory representations may inform machine learning and CBR techniques, and ultimately, the expression of creativity in artificial systems.

Acknowledgements

The Lrn2Cre8 project acknowledges financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 610859.

References

1. Bar, M.: The proactive brain: using analogies and associations to generate predictions. Trends in cognitive sciences 11(7), 280–289 (2007)
2. Bartlett, F.C.: Remembering: A study in experimental and social psychology, vol. 14. Cambridge University Press (1995)
3. Boden, M.A.: The creative mind: Myths and mechanisms. Psychology Press (2004)
4. Bregman, A.S.: Auditory scene analysis: The perceptual organization of sound. MIT press (1994)

5. Dowling, W.J.: Scale and contour: Two components of a theory of memory for melodies. Psychological review 85(4), 341 (1978)
6. Fauconnier, G., Turner, M.: The way we think: Conceptual blending and the mind's hidden complexities. Basic Books (2008)
7. Gärdenfors, P.: Conceptual spaces: The geometry of thought. MIT press (2004)
8. Gervás, P.: An expert system for the composition of formal spanish poetry. Knowledge-Based Systems 14(3), 181–188 (2001)
9. Gjerdingen, R.O.: A classic turn of phrase: Music and the psychology of convention. Univ of Pennsylvania Press (1988)
10. Heath, D., Dennis, A., Ventura, D.: Imagining imagination: A computational framework using associative memory models and vector space models. In: Proc. of the 6th International Conference on Computational Creativity. p. 244 (2015)
11. Hofstadter, D.R.: Analogy as the core of cognition. Keith J. Holyoak, and Boicho N. Kokinov. Cambridge MA: The MIT Press/Bradford Book (2001)
12. Huron, D.B.: Sweet anticipation: Music and the psychology of expectation. MIT press (2006)
13. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: Proceedings of the International Conference on Learning Representations (ICLR), (2014)
14. Koestler, A.: The act of creation. Macmillan (1964)
15. Krumhansl, C.L.: Cognitive foundations of musical pitch, vol. 17. Oxford University Press New York (1990)
16. Lerdahl, F.: Tonal pitch space. Oxford University Press (2001)
17. Minsky, M.: A framework for representing knowledge. Tech. rep., Cambridge, MA, USA (1974)
18. Narmour, E.: The analysis and cognition of melodic complexity: The implication-realization model. University of Chicago Press (1992)
19. Oliva, A.: Gist of the scene. Neurobiology of attention 696(64), 251–258 (2005)
20. Oliva, A., Torralba, A.: Building the gist of a scene: The role of global image features in recognition. Progress in brain research 155, 23–36 (2006)
21. Ontañón, S., Plaza, E.: Amalgams: A formal approach for combining multiple case solutions. In: Case-Based Reasoning. Research and Development, pp. 257–271. Springer (2010)
22. Pearce, M.T., Wiggins, G.A.: Evaluating cognitive models of musical composition. In: Cardoso, A., Wiggins, G.A. (eds.) Proc. of the 4th International Joint Workshop on Computational Creativity. pp. 73–80. London (2007)
23. Pearce, M.T., Wiggins, G.A.: Auditory expectation: the information dynamics of music perception and cognition. Topics in cognitive science 4(4), 625–652 (2012)
24. Piaget, J., Cook, M., Norton, W.: The origins of intelligence in children, vol. 8. International Universities Press New York (1952)
25. Ponsford, D., Wiggins, G.A., Mellish, C.: Statistical learning of harmonic movement. Journal of New Music Research 28(2), 150–177 (1999), http://www.soi.city.ac.uk/ geraint/papers/JNMR97.pdf
26. Rensink, R.A., O'Regan, J.K., Clark, J.J.: To see or not to see: The need for attention to perceive changes in scenes. Psychological science 8(5), 368–373 (1997)
27. Schank, R.C., Abelson, R.P.: Scripts, plans, goals, and understanding: An inquiry into human knowledge structures. Psychology Press (2013)
28. Shepard, R.N.: Recognition memory for words, sentences, and pictures. Journal of verbal Learning and verbal Behavior 6(1), 156–163 (1967)
29. Snyder, B.: Music and memory: An introduction. MIT press (2000)
30. Wiggins, G.A.: A preliminary framework for description, analysis and comparison of creative systems. Knowledge-Based Systems 19(7), 449–458 (2006)

Generation of concept-representative symbols
Position paper

João Miguel Cunha, Pedro Martins, Amílcar Cardoso, Penousal Machado

Departamento de Engenharia Informtica da Universidade de Coimbra
{jmacunha,pjmm,amilcar,machado}@dei.uc.pt

Abstract. The visual representation of concepts or ideas through the use of simple shapes has always been explored in the history of Humanity, and it is believed to be the origin of writing. We focus on computational generation of visual symbols to represent concepts. We aim to develop a system that uses background knowledge about the world to find connections among concepts, with the goal of generating symbols for a given concept. We are also interested in exploring the system as an approach to visual dissociation and visual conceptual blending. This has a great potential in the area of Graphic Design as a tool to both stimulate creativity and aid in brainstorming in projects such as logo, pictogram or signage design.

Keywords: Computational creativity, Computational generation, Concept representation, Visual representation

1 Introduction

Creativity can be seen as the ability to create novel ideas by making connections between existing ones. It plays an important role in the area of Graphic Design not only in conceiving new concepts but also in visually representing them.

As far as visual representation of concepts is concerned, humans have been doing it since more than two hundred thousand years ago – take for example cave paintings. These representations vary from being completely pictorial – e.g. pictograms – to more abstract – e.g. ideographs.

The link between the visual representation and the conceptual connections behind it can in fact be observed. Examples of this can be seen by looking at Chinese characters, more specifically at the ones categorised as Ideogrammic Compounds (see Figure 1). These characters can be decomposed into others, whose concepts are semantically related, belonging to the same (or at least similar) conceptual space [6].

Some authors were inspired by this relationship between concepts to their visual representations. One of them was Charles Bliss who developed a communication system composed of several hundreds of ideographs that can be combined to make new ones – *Blissymbols* [1]. In his system the variation in terms of abstraction degree can also be observed (see Figure 2).

本 木 林 森

Fig. 1. Chinese characters for *root*, *tree*, *woods* and *forest* (left to right). *Root* can be obtained by adding a line to the *tree* character; *woods* character can be obtained (barely) by using two *tree* characters; *woods* can be obtained by using three *tree* characters.

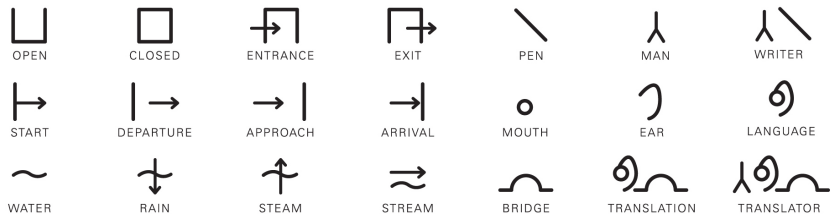


Fig. 2. Blissymbols. Several interesting things can be observed by looking at blissymbols: such as a variation in terms of abstraction degree (there are both pictorial and abstract symbols); by combining symbols, new meanings are obtained (examples in Figure 2: *pen* + *man* = *writer*, *mouth* + *ear* = *language*); by using the same symbols in a different position, a new meaning is obtained (see symbols *water*/*rain*/*steam*/*stream*).

Inspired by examples as the ones presented above, our goal is to conceive an approach for computationally generating concept-representative symbols, i.e. visual representations of concepts. In this paper we present some of the key aspects that have to be considered when generating such symbols and the strategies to explore in order to achieve our objective.

2 Generation of concept-representative symbols

The idea of creating a symbol for a given concept based on its connections to other concepts is, just by itself, interesting. However, if we consider the exploration of this idea using computational means to automate the generation process of the symbols, its potential greatly increases.

We can think of a tool capable of generating symbols whose visual properties would be the outcome of an analysis of the conceptual space of the introduced concept. We believe that such a tool could assist the designer during the ideation process by stimulating its creativity, aiding in brainstorming activities and thus giving rise to new ideas and concepts.

Concerning the visual qualities of the generated symbols, it is crucial to consider several aspects. The first one is the degree of abstraction. This aspect can be considered to be influenced by the choice of the connections used in the symbol generation. Take for example the concept *car*: if we consider the connections between *car* and the concepts *door*, *window* and *wheel*, the resultant symbol will probably be highly pictorial; if we choose to ignore those connections,

the resulting symbol might be more abstract. Ideally, the tool should allow the abstraction degree to be set according to the user's needs.

However, this is not the only aspect that is greatly dependent on the connections used. As observed in the blissymbols, a given combination of symbols might lead to different interpretations. If some perceptual aspects are not considered, this might result in a conflict between the concept and the perception of the symbol. In the next subsection some of these aspects will be presented.

2.1 Considering visual characteristics

When dealing with symbols, it is important to bear in mind some aspects of how a representation's meaning can be changed by changing some of its visual characteristics. The following aspects are essential: position, colour and shape.

The first semiotic aspect – position – can be seen in Figure 2. By putting an arrow next to the *water* symbol a new concept is represented. The concept also varies according to positioning of the arrow. Such details must be considered and a mechanism for analysing them needs to be developed (similar issues have been considered in [2]).

Fig. 3. Left side is shown how the meaning of a banana can change with its colour (mature, green and red banana). Right side is Kiki/Bouba example. Accordingly to Ramachandran 98% of all respondents atribute the name Kiki to the shape on the left and Bouba to the right one, despite having no meaning at all [9]. *Best viewed in colour.*

Another aspect to be regarded is colour. Through a brief analysis of the banana example in Figure 3 it is easy to understand the importance of this aspect. By simply attributing a different colour to the same symbol, its perceptual meaning also changes. In addition, the use of colour has already been proven as a mean of facilitating the interpretation of visual representations of concepts (e.g. [7]). However, its incorrect use has the opposite effect (e.g. Stroop effect), causing interference in its interpretation. On the other hand, a mechanism to avoid an over-use of colour will probably be needed as colour might not be necessary in some symbols.

The third important aspect is shape. When generating visual symbols from textual data (e.g. semantic networks), one cannot avoid dealing with shape. The choice of shape for a given concept is not easy by itself but one has also to consider its visual qualities. Take for example the shapes presented in Kiki/bouba example in Figure 3. Despite not having any meaning at all, there is a clear tendency or bias when attributing names to them. This can be explained as follows, humans

tend to perform mappings among domains, namely between image and sound, as such sharp shapes tend to be associated with sharp sounds and organic shapes with smooth ones [9].

As we have already mentioned, these semiotic aspects have to be considered when generating symbols. This is only possible to achieve by thoroughly analysing the conceptual network and also considering previously generated symbols as both examples and base for the generation of the new symbol.

2.2 Getting information

An extensive analysis of the conceptual space is important but there is another issue that has to be resolved: if the system does not have access to a large source of knowledge – with information about visual properties – it will be difficult, if not impossible, to achieve good results. One possibility is to use an already built semantic network of common sense knowledge (e.g.[8] or [3]).

However, as our main objective is to generate visual representations, knowledge about visual characteristics is required. For that reason, a methodology has to be conceived for acquiring such data. A possible solution is to use a similar approach to the one used by Open Mind Common Sense project – a knowledge acquisition system designed to acquire common sense knowledge from the general public over the web [10]. Our goal is to focus on gathering information about objects' visual characteristics such as colour, shape and texture. These will likely allow us to attain adequate results in terms of symbol generation.

Crowdsourcing will probably be used in our knowledge-gathering process as it easily allows to reach a high number of contributors at a reduced cost. In addition, the validity of online crowdsourced experiments on visual properties and graphical perception has already been demonstrated (e.g. [5]).

This *distributed human project* approach allows us not only to gather data at a scale that would not be possible otherwise but also enables us to study the role of context in perception – one of our goals is to test whether the symbols generated differ accordingly to the location where the data was gathered from.

We also intend to explore other alternatives for populating our semantic network, such as automatic gathering of information. Using Google Image Search is one example of this and can be used to find images related to the content being analysed and consequently extracting useful information from them (e.g. [7]).

2.3 Generating symbols

In our opinion there are, at least, two different ways of generating a symbol for a given concept: (1) starting with no prior knowledge and analysing the conceptual space in order to extract possible visual features to be used in the symbol; (2) using prior knowledge – previously generated symbols of concepts that are, in some way, related to the introduced concept. This would lead to a higher coherence among generated symbols. In both cases, not only direct conceptual connections are used but also more uncommon ones – through a process of analogy. As such,

we argue that mechanisms such as case-based reasoning or conceptual blending [4] are suitable strategies to generate symbols of concepts. As for the former, we can consider a case-base comprised of symbols such as the ones depicted in Figure 2 and develop a system to produce novel ones using analogues to the previous ones. Regarding conceptual blending, the idea is to explore the structure mapping approach to analogy and concept integration based on conceptual spaces and semiotic systems.

3 Conclusion

In this paper we presented our approach to computational generation of concept-representative symbols. We aim to develop a design aiding tool that combines the exploration of conceptual spaces in combination with processes of analogy-making and semiotic analysis to generate possible visual (abstract/semi-abstract) representations for the concepts introduced by the user. We believe that it will help the designer during the ideation process by stimulating its creativity, aiding in brainstorming activities and thus giving rise to new ideas and concepts.

Acknowledgements

The authors acknowledge the financial support from the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under the ConCreTe FET-Open project (grant number 611733) and the PROSECCO FET-Proactive project (grant number 600653).

References

1. Bliss, C.K.: Semantography (Blissymbolics): A Logical Writing for an illogical World. Semantography Blissymbolics Publ (1965)
2. Confalonieri, R., et al.: Using argumentation to evaluate concept blends in combinatorial creativity paper type: Study paper. (to Appear in ICCC 2015)
3. De Smedt, T.: Modeling Creativity: Case Studies in Python. University Press Antwerp (2013)
4. Fauconnier, G., Turner, M.: Conceptual integration networks. Cognitive science 22(2), 133–187 (1998)
5. Heer, J., Bostock, M.: Crowdsourcing graphical perception: using mechanical turk to assess visualization design. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 203–212. ACM (2010)
6. Hew, S., et al.: Using combining evolution of pictogram chinese characters to represent ideogrammic compounds chinese characters. In: Computing and Convergence Technology, 2012 7th International Conference on. pp. 219–223. IEEE (2012)
7. Lin, S., et al.: Selecting semantically-resonant colors for data visualization. In: Computer Graphics Forum. vol. 32, pp. 401–410. Wiley Online Library (2013)
8. Liu, H., Singh, P.: Conceptneta practical commonsense reasoning tool-kit. BT technology journal 22(4), 211–226 (2004)
9. Ramachandran, V.S., Hubbard, E.M.: Synaesthesia–a window into perception, thought and language. Journal of consciousness studies 8(12), 3–34 (2001)
10. Singh, P., et al.: Open mind common sense: Knowledge acquisition from the general public. In: On the move to meaningful internet systems 2002: Coopis, doa, and odbase, pp. 1223–1237. Springer (2002)

Case-Based Reasoning in Health Sciences

Workshop at the
Twenty-Third International Conference on
Case-Based Reasoning
(ICCBR 2015)

Frankfurt, Germany
September 2015

Isabelle Bichindaritz, Cindy Marling, and Stefania Montani
(Editors)

Program Chairs

Isabelle Bichindaritz	State University of New York at Oswego, USA
Cindy Marling	Ohio University, USA
Stefania Montani	University of Piemonte Orientale, Italy

Program Committee

Agnar Aamodt	NTNU, Norway
Juan Manuel Cortado	University of Salamanca, Spain
Peter Funk	Malardalen University, Sweden
Jean Lieber	Loria, University of Nancy, France
Amedeo Napoli	Loria, University of Nancy, France
Lucia Sacchi	University of Pavia, Italy
Rainer Schmidt	Institute for Medical Informatics and Biometry, University of Rostock, Germany

Preface

The community working on health sciences applications of case-based reasoning (CBR) meets again at the International Conference on Case-based Reasoning (ICCBR) this year to share ideas and system descriptions collected in the proceedings of this workshop. This event is the tenth in a series of successful workshops, co- located with different ICCBR/ECCBR conferences. The first nine were held at ICCBR-03, in Trondheim, Norway, at ECCBR-04, in Madrid, Spain, at ICCBR-05, in Chicago, USA, at ECCBR-06 in Olüdeniz, Turkey, at ICCBR-07 in Belfast, Northern Ireland , at ECCBR-08 in Trier, Germany, at ICCBR-09 in Seattle, USA, at ICCBR- 2012 in Lyon, France, and at ICCBR-2013 in Saratoga Springs, USA.

Three papers and one invited speaker summary have been selected this year for presentation during the ICCBR workshops and inclusion in the Workshops Proceedings. The first paper in these proceedings deals with medical process management and presents a tool capable of predicting the evolution of a current process based on previous traces, which can be very useful to ensure compliance with clinical guidelines [Bottrighi et al.]. The second paper highlights how a case-based approach to data analysis can aid in integrating new fitness band data into machine learning models for blood glucose prediction [Marling et al.]. The third paper focuses on the importance and variety of data mining methods in case-based reasoning, in particular for health sciences applications [Bichindaritz]. Finally, the invited speaker summary discusses how hybrid case-based reasoning is likely to change the future of health science and healthcare [Funk].

These papers report on the research and experience of 11 authors working in three different countries on a wide range of problems and projects, and illustrate some of the major trends of current research in the area. Overall, they represent an excellent sample of the most recent advances of CBR in the health sciences, and promise very interesting discussions and interaction among the major contributors in this niche of CBR research.

September 2015
Frankfurt

Isabelle Bichindaritz
Cindy Marling
Stefania Montani

Trace Retrieval as a Tool for Operational Support in Medical Process Management

Alessio Bottrighi (1), Luca Canensi (2), Giorgio Leonardi (1), Stefania Montani (1), Paolo Terenziani (1)

(1) DISIT, Computer Science Institute, Università del Piemonte Orientale, Alessandria, Italy
(2) Department of Computer Science, Università di Torino, Italy

Abstract. Operational support assists users while process instances are being executed, by making predictions about the instance completion, or recommending suitable actions, resources or routing decisions, on the basis of the already completed process traces. Operational support can be particularly useful is the case of medical processes, where a given process instance execution may differ from the indications of the existing reference clinical guideline. In this paper, we propose a Case Based Reasoning approach to medical process management operational support. The framework enables the user to retrieve past traces by submitting queries representing complex patterns exhibited by the current process instance. Information extracted from the retrieved traces can guide the medical expert in managing the current instance in real time. The tool relies on a tree structure, allowing fast retrieval from the available event log. Thanks to its characteristics and methodological solutions, the tool implements operational support tasks in a flexible, efficient and user friendly way.

1 Introduction

Operational support is a process management activity meant to assist users while process instances are being executed, by making predictions about the instance completion, or recommending suitable actions, resources or routing decisions, on the basis of the already completed instances [1]. Operational support can be particularly useful in the case of medical processes, where a given process instance execution may (significantly) differ from the indications of the existing reference clinical guideline. Indeed, specific patient characteristics (e.g., co-morbidities, allergies, etc.), or local resource constraints, may lead to deviations from the default behavior, which need to be managed in real time. Prediction and recommendation heavily rely on experiential knowledge, stored in the so-called "event log" in the form of past process traces. Case Based Reasoning (CBR) [2], and specifically the retrieval step in the CBR cycle, thus appears to be a very valuable methodology for implementing these operational support tasks. The percentage of retrieved traces that, e.g., were completed on time, can then be used to calculate the probability that the current instance will complete on time too. A

similar approach can be adopted to estimate costs, or predict problems. Moreover, the best actions to execute next can also be extracted from the retrieved traces.

In this paper we propose a case-based retrieval framework, where cases are traces of process execution, aimed at enabling prediction and recommendation in medical process operational support. In our framework, queries can be composed of several simple patterns (i.e., single actions, or direct sequences of actions), separated by delays (i.e., interleaved actions we do not care about). Delays can also be imprecise (i.e., the number of interleaved actions can be given as a range). To the best of our knowledge, an operational support facility like this is not available in the tools described in the literature. Our framework relies on a tree structure, called the *trace tree*, allowing fast retrieval, thus avoiding a flat search for all the traces in the log that match the input pattern. The trace tree is a sort of "model" of the traces, that we learn using a process mining technique we recently implemented [3], and built in such way that it can be used as an index[1].

The paper is organized as follows. In section 2 we illustrate our retrieval approach. In section 3 we discuss related work. In section 4 we present our concluding remarks and future work directions.

2 Trace retrieval

In our framework, trace retrieval relies on a tree structure, called the **trace tree**, in order to avoid a flat search for all the traces in the log that match the input query. In the following, we will first describe the trace tree semantics, and then introduce our query language and, finally, our retrieval procedure.

Trace tree semantics. In the trace tree, nodes represent actions, and arcs represent a precedence relation between them. More precisely, each node is represented as a pair $< P, T >$.

P denotes a (possibly unary) set of actions; actions in the same node are in AND relation, or, more properly, may occur in any order with respect to each other. Note that, in such a way, each path from the starting node of the tree to a given node N denotes a set of possible process patterns (called *support patterns* of N henceforth), obtained by following the order represented by the arcs in the path to visit the trace tree, and ordering in each possible way the actions in each node (for instance, the path $\{A, B\} \rightarrow \{C\}$ represents the support patterns "ABC" and "BAC").

T represents a set of pointers to all and only those traces in the log whose prefixes exactly match one of the patterns in P (called *support traces* henceforth). Specifically, prefixes correspond to the entire traces if the node at hand is a leaf. In the case of a node representing a set of actions to be executed in any order, T is

[1] While the motivations for defining such a novel mining algorithm, and its advantages with respect to existing process mining literature contributions (e.g., ProM [4]), are extensively discussed elsewhere [3], in this work we focus on its usage to support case retrieval.

more precisely composed of several sets of support traces, each one corresponding to a possible action permutation.

Since all traces start with a dummy common action #, the root node contains the action #, paired to the pointers to all log traces.

Query language. In a tool implementing this framework, the user can issue a query, composed of one or more simple patterns to be searched for. In turn, simple patterns are defined as one or more actions in direct sequence. Multiple simple patterns can be combined in a complex pattern, by separating them by delays. A delay is a sequence of actions interleaved between two simple patterns; the semantics is that we do not care about these actions, so they will not be specified in the query. Instead, only their number will be provided, possibly in an imprecise way (i.e., we allow the user to express the number of interleaved actions as a range).

Formally, a query is represented in the following format:

$\langle (min_1, max_1)SP_1(min_2, max_2)SP_2...(min_k, max_k)SP_k(min_{k+1}, max_{k+1})\rangle$

where:

- SP_j is a simple pattern (i.e. a sequence of letters, representing the actions we are looking for; these actions have to be in direct sequence);
- (min_j, max_j) is the delay between two items (i.e., two simple patterns, or a simple pattern and the trace starting/ending point), expressed as a range in the number of interleaved actions.

As an example, the query
$\langle (0,0)B(0,1)E(2,2)Z(0,1)\rangle$
looks for action B, which has to start at the very beginning of the trace (just after the start action # - all traces start with a dummy common action # in our approach). This first simple pattern B must be followed (with zero or a single interleaved action in between) by action E. E must be followed by two actions, which we do not care about; after them, Z is required. Z can be the final action, or can be followed by one additional action we do not care about.

For instance, in the stroke management domain, where we will test our approach, actions B, E and Z could correspond to "Arrival at the emergency department", "Neurological examination", and "Chest X-ray" respectively. Looking for "Arrival at the emergency department" at the very beginning of the trace allows the exclusion of all those patients that were first stabilized at home or in the ambulance. The query then aims for searching for those situations where "Neurological examination" is executed early, and before "Chest X-ray"; in fact, this specific ordering is not mandatory, because "Chest X-ray" is a procedure common to many different disease management processes, and may be executed at different times, also depending on the availability of the X-ray machine. Similarly, in some cases "Neurological examination" might be delayed, if the neurologist is not available. The two actions between "Neurological examination" and "Chest X-ray" would typically correspond to "CTA" and "ECG", always obtained to every patient in the case of a suspected stroke (but not explicitly queried in the example).

It is worth noting that a query written as above corresponds to a whole set of queries, each one obtained by choosing a specific delay value and specific actions in each of the (min_j, max_j) intervals. Every query in this set can be made partially explicit as a string, containing as many dummy symbols $*$ as needed, to cover the corresponding delay length (where the dummy symbol is chosen because we are not interested in the specific interleaved actions). For example, the query above would correspond to the following four partially explicit queries, whose length ranges from 6 to 8 actions (including #), where the dummy symbol $*$ has been properly inserted, according to the delay values information: $\#BE**Z$; $\#BE**Z*$; $\#B*E**Z$; $\#B*E**Z*$

Since each $*$ could be substituted by any of the N types of actions recorded in the log and/or existing in the application domain, the example query corresponds to $N^2 + 2 * N^3 + N^4$ totally explicit queries.

The problem is obviously combinatorial, with respect to the possible delay ranges and action types. We thus believe that extensional approaches (in which only explicit queries can be issued) would not be feasible in many domains. Our query language, allowing for compact "intensional" queries, is therefore a significant move in the direction of implementing an efficient and user-friendly operational support tool.

Trace retrieval. In order to retrieve the log traces that match a query, we have implemented a multi-step procedure, articulated as follows: (1) automaton generation; (2) tree search; (3) filtering.

To generate the automaton, in turn, we implement the following procedure:

1. transform the query into a regular expression;
2. apply the Berry and Sethi [5] algorithm, to build a non-deterministic automaton that recognizes the regular expression above;
3. unfold the non-deterministic automaton;
4. transform the unfolded non-deterministic automaton into a deterministic automaton [6].

Steps (1) and (4) are trivial. As regards step (1) note that our query language is just a variation of regular expressions, useful to express delays and "do not care" (i.e., dummy) symbols in a compact way. The cost of step (1) is linear in the number of delays used in the query. Steps (2) and (3) use classical algorithms in the area of formal languages. The cost of step (2) is linear in the number of symbols in the query expressed as a regular expression (i.e., the output of step (1) [5]), and the cost of step (3) is the product between the number of dummy symbols in the query and the cardinality of the action symbols available in the log. Step (4) substitutes each arc labeled by the dummy symbol in the automaton with a set of arcs, one for each action in the event log. Although in the worst case step (4) is exponential with respect to the number of states in the automaton (i.e., the output of step (2)), note that the worst case is rare in practice [7].

Once the deterministic automaton has been obtained, it would be possible to exploit it in a classical way, by providing all event log traces in input to it, to verify which of them match the query. However, some of these traces may be identical, or share common prefixes of various length, so that the straightforward

approach would lead to repeated analyses of the common parts. In order to optimize efficiency, we have therefore proposed a novel approach, that provides the trace tree as an input to the automaton. Each path in the trace tree may index several identical support traces, that will be considered only once, thus speeding up retrieval with respect to a flat search into the event log. Moreover, in the tree common prefixes of different traces are represented just once, as common branches close to the root (different postfixes can then stem from the common branches, to reach the various leaves). These common parts will be executed on the automaton only once, without requiring repeated, identical checks.

It is worth noting that providing a tree as an input to the automaton represents a significantly novel contribution, since in the formal languages literature the input to be executed on the automaton is typically a string. The work in [8] represents an exception, but the tree it exploits (a Patricia tree) has very different semantics with respect to ours.

In detail, our approach operates as follows: the algorithm *Search_Process* (see algorithm 1) takes in input the trace tree T and the deterministic automaton A, and provides as an output a set of pairs, composed of a trace tree leaf node and a corresponding string. Notably, there could be several pairs having the same leaf node. Each of the strings is an explicit instantiation of the query represented by the automaton, verified by (some of) the support traces in the leaf node. The output support traces are then provided as an input to the filtering step (see below).

Basically, *Search_Process* executes a breadth first visit of the trace tree; it exploits the variable *searching*, defined as a set of triples, composed of a trace tree node, an automaton state, and the string that has been recognized on the automaton so far. Initially (line 4), *searching* contains the root (with the dummy action #), paired to the initial state of the query automaton and to the empty string. The visit procedure (lines 7-35) extracts one triple at a time from the set *searching*. If the *node* in the triple contains a set of actions to be executed in any order (line 9), we simulate all the permutations on the automaton, and save the states we reach and the corresponding recognized strings into *new_states* set (line 12). If the node contains one single action, we simply simulate it on the automaton, and save the state we reach and the corresponding string into *new_states* set (line 17). In both cases, the string saved in *new_states* is the one in the input triple properly updated with the newly recognized symbols.

After the simulation, if the *node* at hand is a leaf (line 20), then for each item in *new_states* we check whether the state component is a final state (lines 22-24); if this is the case, *node* and the string paired to the final state are saved in the output variable *result* (line 23). Otherwise, if *node* is not a leaf, we pair its children to all the items in *new_states*, and save these objects into *searching* (lines 27-33). The visit terminates when *searching* is empty, i.e., all tree levels have been visited. The visit procedure is linear in the number of the trace tree nodes.

Referring to our example query, providing the trace tree in figure 1 as an input to the algorithm *Search_Process*, after examining the root (which is triv-

ALGORITHM 1: Pseudo-code of the procedure *Search_Process*.

```
1  Search_Process(T, A)
2  Output: set of < node, string >
3  result ← {}
4  searching ←< root(T), 0, empty >
5  repeat
6  |  tmp ← {}
7  |  foreach < node, state, string > ∈ searching do
8  |  |  new_states ← {}
9  |  |  if node is an any-order-node then
10 |  |  |  foreach Perm ∈ permutation(node) do
11 |  |  |  |  foreach act ∈ Perm do
12 |  |  |  |  |  new_states ← new_states ∪ simulate(A, act,state,string)
13 |  |  |  |  end
14 |  |  |  end
15 |  |  end
16 |  |  else
17 |  |  |  new_states ← simulate(A, action(node),state,string)
18 |  |  end
19 |  |  if new_states ≠ {} then
20 |  |  |  if node is a leaf then
21 |  |  |  |  foreach < state, string > ∈ new_states do
22 |  |  |  |  |  if final(state) then
23 |  |  |  |  |  |  result ← result ∪ < node, string >
24 |  |  |  |  |  end
25 |  |  |  |  end
26 |  |  |  end
27 |  |  |  else
28 |  |  |  |  foreach n ∈ sons(node) do
29 |  |  |  |  |  foreach < state, string > ∈ new_states do
30 |  |  |  |  |  |  tmp ← tmp ∪ < n, state, string >
31 |  |  |  |  |  end
32 |  |  |  |  end
33 |  |  |  end
34 |  |  end
35 |  end
36 |  searching ← tmp
37 until searching ≠ {}
38 return result
```

ial), *searching* contains the children of the root A, B, and C, paired with the state of the deterministic automaton, and with the string #. We simulate the actions A, B, and C on the automaton. Only B (i.e., "Arrival at the emergency department") is recognized, generating a state saved in *new_states* with the corresponding string $\#B$ (line 17). We then pair the children of node B (E, D, $D-E$) to the item in *new_states* and save these triples into *searching* (lines 27-33). In the stroke management domain, E corresponds to "Neurological examination" and D to "CTA". Continuing the visit, particularly interesting is the case of node $D-E$, which requires consideration of all the possible permutations of actions D and E. Both the permutations DE and ED are initially recognized. However, as the visit proceeds and node $P-Z$ is reached (with P corresponding to "ECG" and Z corresponding to "Chest X-ray"), it turns out that DE must be followed by the permutation PZ to match the query; on the other hand, if the choice ED is made, it must be followed by ZP. Indeed, the query imposes some *constraints that cannot be checked only locally*, i.e., referring to a single node along the branch. After this step of the visit (depth 5 in the tree), the recognized partial strings paired to node $P-Z$ are $\#BDEOPZ$ and $\#BEDOZP$ (with O corresponding to "NMR"). Notably, the patterns $\#BDEOZP$ and $\#BEDOPZ$ do not match the input query.

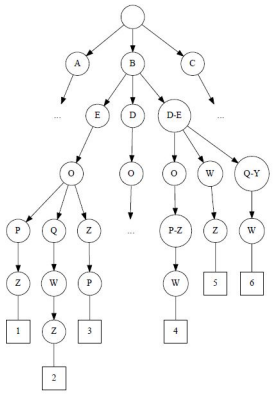


Fig. 1. Trace tree in the example.

If an output leaf node ends a branch which includes one or more nodes with actions to be executed in any order, it is possible that only some of the permutations of these actions are acceptable to answer the query. However, the trace tree leaf node indexes all the traces corresponding to the various support patterns (i.e., considering all possible permutations). Therefore, the support traces must be filtered.

To do so, without the need of operating directly on the input traces, we exploit the fact that, in each node with actions to be executed in any order, every permutation is explicitly stored, and each permutation indexes all and

only the support traces corresponding to it. Thus, the basic idea of our filtering step is simple: for each output pair ⟨ Node, String ⟩ of the tree search step, we navigate the trace tree from *Node* back to the root, maintaining, in each any order node, only the (pointers to) the traces corresponding to *String* (this can be easily done through the operation of intersection between sets of pointers).

The complexity of the filtering step is superiorly limited by the number of ⟨ Node, String ⟩ pairs identified as an output of the tree search step, multiplied by the tree depth.

Obviously, if the leaf node ends a branch that contains no nodes with actions to be executed in any order, the leaf support traces can be directly presented to the user, and the filtering step is not required.

3 Related work

Operational support techniques are implemented in the open source framework ProM [4] (developed at the Eindhoven University of Technology), which represents the state of the art in process mining research. In ProM, prediction and recommendation are typically supported by replaying log traces on the transition system [9], a state-based model that explicitly shows the states a process can be in, and all possible transitions between these states. The replay activity allows calculation of, e.g., the mean time to completion from a given state, or the most probable next action to be executed. In ProM's approach, statistics on event log traces are thus used for operational support, but the overall technique is very different from the one we propose in this paper, and no trace retrieval on the basis of complex pattern search is supported.

On the other hand, traces have been recently considered in the CBR literature, as sources for retrieving and reusing user's experience. As an example, at the International Conference on CBR in 2012, a specific workshop was devoted to this topic [10]. In 2013, Cordier et al. [11] proposed trace-based reasoning, a CBR approach where cases are not explicitly stored in a library, but are implicitly recorded as "episodes" within traces. The elaboration step, in which a case is extracted from a trace, is thus one of the most challenging parts of the reasoning process. Zarka et al. [12] extended that work, and defined a similarity measure to compare episodes extracted from traces. In these works, traces are typically intended as observations captured from users' interaction with a computer system. Trace-based reasoning was exploited in recommender systems [13, 14], and to support the annotation of digitalized cultural heritage documents [15]. Leake used execution traces recording provenance information to improve reasoning and explanation in CBR [16]. In the Phala system [17], the authors supported the generation and composition of scientific workflows by mining execution traces for recommendations to aid workflow authors. Finally, Lanz et al. used annotated traces recorded when a human user played video games in order to feed a case-based planner [18].

All these approaches implement forms of reasoning on traces. However, to the best of our knowledge, a tace-based CBR approach has never been exploited for operational support in Medical Process Management.

4 Conclusions

In this paper, we have introduced a novel framework for trace retrieval, designed to implement operational support tasks in a flexible, efficient and user-friendly way. With respect to existing operational support facilities, our tool is more flexible because it allows to search for traces that exhibit complex query patterns, identified in the input trace. The tool is also efficient and user-friendly, since:

- by allowing for the use of (imprecise) delays in the query language, it enables users to express a very large number of explicit queries in a compact way;
- by providing the trace tree as an input to the automaton:
  - it speeds up retrieval relative to a flat search into the event log;
  - it executes common prefixes of different traces only once on the automaton, avoiding repeated, identical checks.

In the future, we plan to extensively test the overall framework on real world traces, which log the actions executed during stroke patient management in a set of Northern Italy hospitals.

References

1. W. Van der Aalst. *Process Mining. Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
2. A. Aamodt and E. Plaza. Case-based reasoning: foundational issues, methodological variations and systems approaches. *AI Communications*, 7:39–59, 1994.
3. L. Canensi, S. Montani, G. Leonardi, and P. Terenziani. Chapman: a context aware process miner. In *Workshop on Synergies between Case-Based Reasoning and Data Mining, International Conference on Case Based Reasoning (ICCBR)*, 2014.
4. B. van Dongen, A. Alves De Medeiros, H. Verbeek, A. Weijters, and W. Van der Aalst. The proM framework: a new era in process mining tool support. In G. Ciardo and P. Darondeau, editors, *Knowledge Mangement and its Integrative Elements*, pages 444–454. Springer, Berlin, 2005.
5. G. Berry and R. Sethi. From regular expressions to deterministic automata. *Theor. Comput. Sci.*, 48(3):117–126, 1986.
6. M. S. Lam, A. V. Aho, R. Sethi, and J. D. Ullman. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley, 2006.
7. J. van Leeuwen. *Handbook of Theoretical Computer Science*. MIT Press, 1994.
8. Ricardo A. Baeza-Yates and Gaston H. Gonnet. Fast text searching for regular expressions or automaton searching on tries. *J. ACM*, 43(6):915–936, November 1996.
9. B. F. Van Dongen, N. Busi, and G. M. Pinna. An iterative algorithm for applying the theory of regions in process mining.

10. M. W. Floyd, B. Fuchs, P. Gonzalez-Calero, D. Leake, S. Ontanon, E. Plaza, and J. Rubin. *TRUE: Traces for Reusing User's Experiences Cases, Episodes, and Stories, International Conference on Case Based Reasoning (ICCBR)*. Lyon, 2012.
11. Amlie Cordier, Marie Lefevre, Pierre-Antoine Champin, Olivier Georgeon, and Alain Mille. Trace-Based Reasoning — Modeling interaction traces for reasoning on experiences. In *The 26th International FLAIRS Conference*, May 2013.
12. Raafat Zarka, Amlie Cordier, Elod Egyed-Zsigmond, Luc Lamontagne, and Alain Mille. Similarity Measures to Compare Episodes in Modeled Traces. In Springer, editor, *International Case-Based Reasoning Conference (ICCBR 2013)*, Lecture Notes in Computer Science, pages 358–372. Springer Berlin Heidelberg, July 2013.
13. Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Proceedings of the 2008 ACM Conference on Recommender Systems*, RecSys '08, pages 335–336, New York, NY, USA, 2008. ACM.
14. Raafat Zarka, Amélie Cordier, Elöd Egyed-Zsigmond, and Alain Mille. Contextual trace-based video recommendations. In *Proceedings of the 21st International Conference Companion on World Wide Web*, WWW '12 Companion, pages 751–754, New York, NY, USA, 2012. ACM.
15. Reim Doumat, Elöd Egyed-Zsigmond, and Jean-Marie Pinon. User trace-based recommendation system for a digital archive. In *Proceedings of the 18th International Conference on Case-Based Reasoning Research and Development*, ICCBR'10, pages 360–374, Berlin, Heidelberg, 2010. Springer-Verlag.
16. David B. Leake. Case-based reasoning tomorrow: Provenance, the web, and cases in the future of intelligent information processing. In *Intelligent Information Processing V - 6th IFIP TC 12 International Conference, IIP 2010, Manchester, UK, October 13-16, 2010. Proceedings*, page 1, 2010.
17. D. B. Leake and J. Kendall-Morwick. Towards case-based support for e-science workflow generation by mining provenance. In K.D. Althoff, R. Bergmann, M. Minor, and A. Hanft, editors, *Proc. ECCBR 2008, Advances in Case-Based Reasoning*, volume 5239 of *Lecture Notes in Computer Science*, pages 269–283. Springer, 2008.
18. A. Lanz, B. Weber, and M. Reichert. Workflow time patterns for process-aware information systems. In *Proc. BMMDS/EMMSAD*, pages 94–107, 2010.

Case-Based Reasoning as a Prelude to Big Data Analysis: A Case Study

Cindy Marling[1], Razvan Bunescu[1],
Babak Baradar-Bokaie[2] and Frank Schwartz[2]

[1] School of Electrical Engineering and Computer Science
Russ College of Engineering and Technology
Ohio University, Athens, Ohio 45701, USA
marling@ohio.edu, bunescu@ohio.edu

[2] Department of Specialty Medicine
Heritage College of Osteopathic Medicine
Ohio University, Athens, Ohio 45701, USA
babak.bokaie@gmail.com, schwartf@ohio.edu

Abstract. The 4 Diabetes Support System (4DSS) is a prototypical hybrid case-based reasoning (CBR) system that aims to help patients with type 1 diabetes on insulin pump therapy achieve and maintain good blood glucose control. The CBR cycle revolves around treating blood glucose control problems by retrieving and reusing therapeutic adjustments that have been effectively used to treat similar problems in the past. Other artificial intelligence (AI) approaches have been integrated primarily to aid in situation assessment: knowing when a patient has a blood glucose control problem and characterizing the type of problem that the patient has. Over the course of ten years, emphasis has shifted toward situation assessment and machine learning approaches for predicting blood glucose levels, as that is the area of greatest patient need. The goal has been to make large volumes of raw insulin, blood glucose and life-event data actionable. During the past year, newly available fitness bands have provided a potentially valuable source of additional data for controlling diabetes. Because it was initially unclear whether or how this new data might be leveraged, a case study was conducted, and CBR was once again called into play. This paper describes the case study and discusses the potential of CBR to serve as a prelude to big data analysis.

1 Introduction

The World Health Organization estimates that there are 347 million people living with diabetes [11]. From 5 to 10% of them have type 1 diabetes (T1D), the most severe form, in which the pancreas fails to produce insulin. T1D is neither curable nor preventable; however, it can be treated with insulin and effectively managed through blood glucose (BG) control. Good BG control helps to delay or prevent long-term diabetic complications, including blindness, amputations,

kidney failure, strokes, and heart attacks [4]. Therefore, it is important to rapidly identify and correct BG control problems.

The 4 Diabetes Support System (4DSS) is a prototypical hybrid case-based reasoning (CBR) system that detects and predicts BG control problems and suggests personalized therapeutic adjustments to correct them. The 4DSS has been extensively described in the literature [6, 7, 9]; a brief system overview is presented in the next section. A critical research thrust that grew out of work on the 4DSS is how to continuously, in real-time, predict that a BG control problem is about to occur. The key is to be able to accurately predict what the BG level will be in the next 30 to 60 minutes, which would allow enough time to intervene and prevent predicted problems. Large volumes of raw insulin and BG data are available for analysis. Machine learning algorithms for time series prediction have been efficaciously applied. Studies conducted on retrospective data show that our system predicts BG levels comparably to physicians specializing in diabetes care, but not yet well enough for use by patients in the real world [2, 8].

Recently, commercially available fitness bands and smart watches, such as the Basis Peak, Nike Fuelband, Fitbit, and Apple Watch, have made it practical to inexpensively and unobtrusively collect large quantities of physiological data. As this data may be indicative of patient activity impacting BG levels, it could potentially be used to improve BG level prediction. However, due to the complicated nature of the problem, it was not initially clear whether or how this data could be leveraged. Therefore, a case study was conducted in which a patient with T1D wore a fitness band in addition to his usual medical devices for two months. The data was consolidated and displayed via custom visualization software. The patient, his physician, and artificial intelligence (AI) researchers met weekly to review and interpret the data, using a protocol like that employed to build the 4DSS. This case-based focus shed light on how the new data could be integrated into machine learning models and leveraged to improve BG prediction. We posit that a case-based approach is especially useful in dealing with new data sources, new patients, and new medical conditions, and that early lessons learned through the CBR process can aid in later big data analysis.

2 Background

This section briefly describes the 4DSS and the work on machine learning models for blood glucose prediction prior to the new case study.

2.1 The 4 Diabetes Support System

A graphical overview of the prototypical hybrid CBR system is shown in Figure 1. The patient provides BG, insulin and life-event data to the system. BG and insulin data are uploaded from the patient's prescribed medical devices. The patient enters data about life events that impact BG levels, such as food, exercise, sleep, work, stress and illness, using a smart phone. The data is scanned by

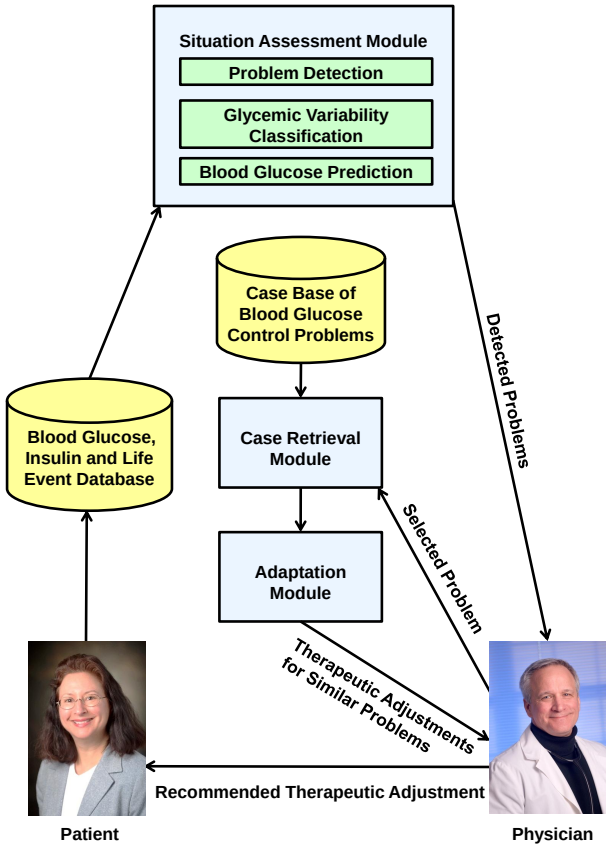


Fig. 1. Overview of the 4 Diabetes Support System, reproduced from [6]

the situation assessment module, which detects and predicts BG control problems. The most critical types of problems are: hyperglycemia, or high BG, which contributes to long-term diabetic complications; and hypoglycemia, or low BG, which may result in severe immediate reactions, including weakness, dizziness, seizure or coma. The situation assessment module reports detected problems to the physician. The physician selects a problem of interest, which is then used by the case retrieval module to obtain the most similar case or cases from the case base. Each retrieved case contains a specific BG control problem experienced by a T1D patient, a physician's recommended therapeutic adjustment, and the clinical outcome for the patient after making the therapeutic adjustment. Retrieved cases go to the adaptation module, which personalizes a retrieved solution to fit the specific needs of the current patient. A solution is a therapeutic adjustment comprising one or more actions that a patient can take. Adapted therapeutic adjustments are displayed to the physician as decision support. The physician

decides whether or not to recommend the therapeutic adjustments to the patient. Directly providing suggestions to the patient, while a long-term goal, would require regulatory approval.

2.2 Machine Learning Models for Blood Glucose Prediction

Originally conceived of as important for situation assessment, BG prediction has multiple potential applications that could enhance safety and quality of life for people with T1D if incorporated into medical devices. These applications include: alerts to warn of imminent problems; decision support for taking actions to prevent impending problems; "what if" analysis to project the effects of lifestyle choices on BG levels; and integration with closed-loop control algorithms for insulin pumps (aka the "artificial pancreas"). Predicting hypoglycemia is especially important, both for patient safety and because hypoglycemia is a limiting factor for intensive insulin therapy [3].

In our BG prediction approach, a generic physiological model is used to generate informative features for a Support Vector Regression (SVR) model that is trained on patient specific data. The physiological model characterizes the overall dynamics into three compartments: meal absorption, insulin dynamics, and glucose dynamics. The parameters of the physiological model are tuned to match published data and feedback from physicians. To account for the noise inherent in the data, the state transition equations underlying the continuous dynamic model are incorporated in an extended Kalman filter.

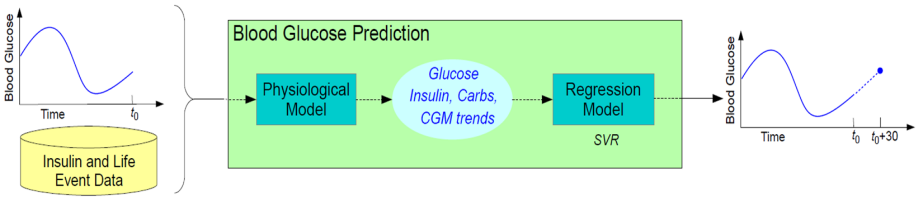


Fig. 2. Overview of the Blood Glucose Level Prediction Process

Figure 2 shows the overall BG level prediction process. A continuous dynamical system implementing the set of physiological equations is run in prediction mode for 30 and 60 minutes. Physiological model predictions are then used as features for an SVR model that is trained on the two weeks of data preceding the test point. Furthermore, an AutoRegressive Integrated Moving Average (ARIMA) model is trained on the same data and its predictions are used as additional features. The models are trained to minimize Root Mean Square Error (RMSE). SVR predictions are made at 30 and 60 minute intervals and compared to BG levels at prediction time (t_0), ARIMA predictions, and predictions made by physicians specializing in diabetes care. Results are shown in Figure 3.

Horizon	t_0	ARIMA	Physician	SVR
30 Min	27.5	22.9	**19.8**	**19.5**
60 Min	43.8	42.2	**38.4**	**35.7**

Fig. 3. RMSE of the Best SVR Model vs. the t_0, ARIMA, and Physician Predictions

3 The Case Study

The recent proliferation of commercially available fitness bands provides an opportunity to exploit new data from inexpensive, unobtrusive, portable physiological sensors. These sensors provide signals indicative of activities that are known to impact BG levels, including sleep, exercise and stress. The hope is that, by incorporating these signals, we can obtain a more accurate picture of patient activity, while reducing or eliminating the need for the patient to self-report life events. We conducted an N-of-1 study in order to learn whether or how this influx of new data could be leveraged to advantage.

The subject was a middle-aged physician who has had T1D since childhood. For two months, he wore a fitness band along with his regularly prescribed medical devices and entered life-event data via his smart phone. The fitness band, a Basis Peak, provided data for galvanic skin response (GSR), heart rate, and skin and air temperatures. The medical devices, a Medtronic insulin pump and a Dexcom continuous glucose monitoring (CGM) system, provided insulin and BG data. All of this data was consolidated in the 4DSS database.

Once a week, the subject met with his physician and AI researchers to review and analyze the data. The consolidated data was displayed via custom-built visualization software called PhysioGraph. A screen shot from PhysioGraph, showing the different types of data, is shown in Figure 4. BG control problems identified by the 4DSS software, the subject, and/or his physician, were visualized and discussed. We looked for visual patterns in the fitness band data during the times when the problems occurred.

Preliminary findings based on these visualizations were encouraging. While even subtle patterns may be detected by machine learning algorithms, we were able to detect some marked patterns as humans. The most pronounced pattern was a rise in GSR with severe hypoglycemia. The most interesting pattern revolved around shoveling snow. The study was conducted during an unusually harsh winter in which the patient (and most of the rest of us) had to frequently shovel heavy snow. Shoveling snow is strenuous exercise, and exercise is known to lower BG levels. After shoveling for extended periods, the subject sometimes experienced hypoglycemia. This is a problem we would like to predict, because, if alerted, the patient could take action to prevent it. There was a discernable pattern in the fitness band data surrounding this problem. GSR and heart rate rose, while skin and air temperature dropped. While we do not yet know if this

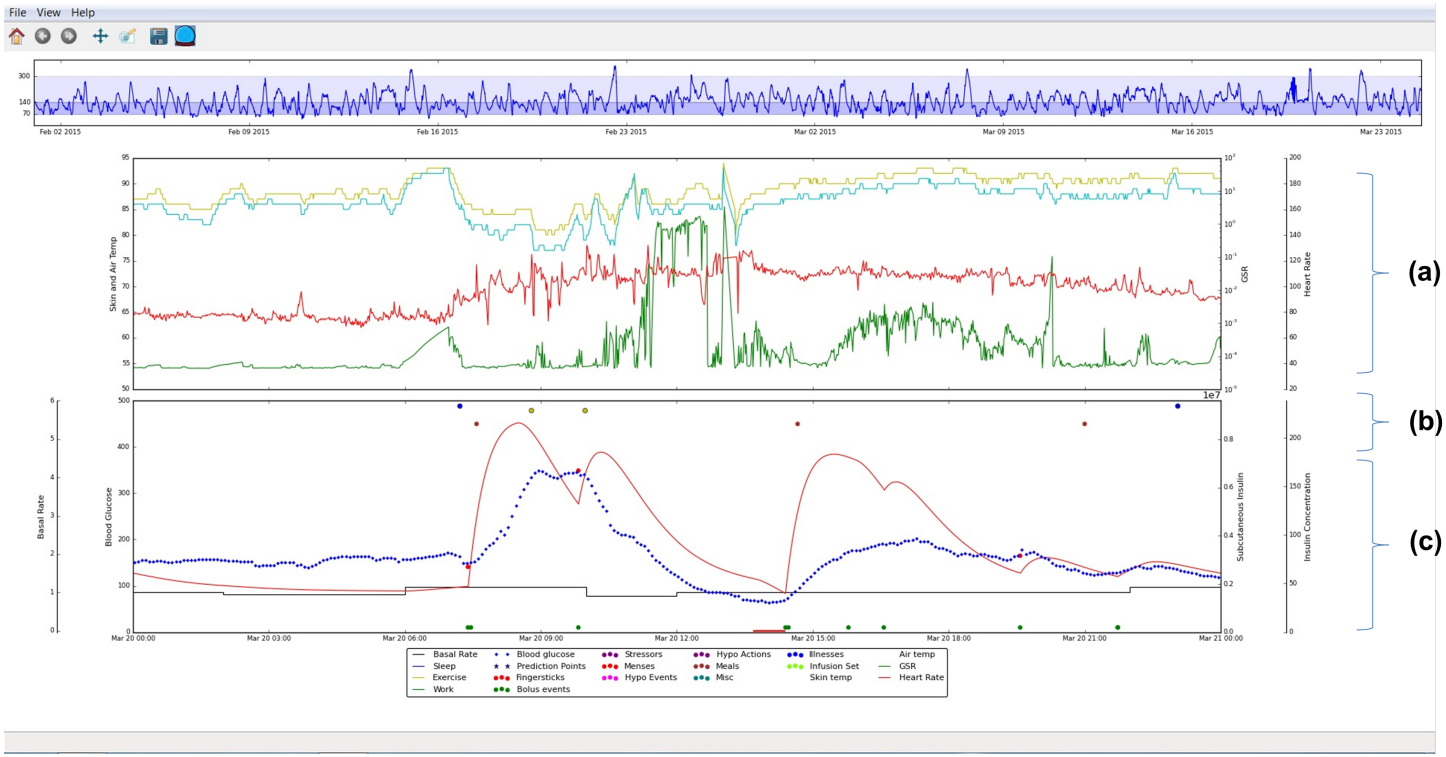


Fig. 4. A Screen Shot from PhysioGraph. Data from the fitness band is displayed in section (a). This includes GSR (green), heart rate (red), skin temperature (gold) and air temperature (cyan). Icons indicating life events appear in section (b). These icons are clickable in PhysioGraph to view event details. BG data and insulin data are shown in section (c). The dotted curve (blue) represents CGM data, while the solid curve (red) models the total insulin in the patient's system.

combination of signals will allow us to predict hypoglycemia or only detect it, we now have leads to follow.

4 Discussion

When work began on the 4DSS in 2004, we had little data and no structured cases. As we built our case base and collected data from over 50 T1D patients, we developed a database that enabled us to build machine learning models for purposes we did not envision in 2004. With more data, it becomes possible to leverage more AI approaches for more purposes. When data or knowledge is limited, however, CBR can be an enabling approach. As non-health-related examples, we can think of leveraging all of the information possible from a single oil spill, or of basing product recommendations for a customer with no purchase history on what similar customers have bought. In the medical arena, CBR has also proven useful for dealing with new situations. For example, CARE-PARTNER used CBR to determine appropriate follow-up care for the earliest stem cell transplant patients [1]. Once many patients had undergone stem cell transplantation and received follow-up care, their collective experiences were distilled into clinical practice guidelines that were used in lieu of CBR. Today, nearly 20 years later, big data tools like IBM Watson Health [5] may allow us to further evaluate, refine, and personalize treatment for these patients.

In the diabetes domain, big data is not yet publicly available; however, we anticipate its near-term future availability. Three non-technical factors contribute to this lack of data: (1) most diabetes patients do not yet wear devices or use systems that continuously collect data; (2) medical device manufacturers do not yet allow access to raw data in real-time, but require the use of their own proprietary software; and (3) patient privacy concerns inhibit data sharing, even when data exists. There has been a recent drive to collect and consolidate data from all T1D patients and all types of (currently incompatible) medical devices, spearheaded by the non-profit organization Tidepool [10]. A goal is to be able to analyze and leverage continuous data from hundreds of thousands of patients to improve diabetes care and outcomes for individuals. Our case engineering, based on the limited data we have already collected, could serve to jump start such efforts.

At the heart of any CBR system is the case. The case is a knowledge structure that, especially in complex medical domains, may embody more than a collection of readily available feature-value pairs. The design of a case for a CBR system begins with the analysis of real-world cases to identify problems, solutions and outcomes. It is necessary to understand and define the features that make cases similar to each other, reusable in different circumstances, and adaptable to the case at hand. Features engineered for cases may provide machine learning algorithms with better inputs than raw data or surface features.

In our case study, the fitness band provided physiological signals that were not a part of our original case design. There were 20 times as many data points per patient per day as we had previously collected. We did not know how the

new data could be used to anticipate or detect blood glucose control problems or, more fundamentally, how it related to BG levels in people with T1D. The inputs to our SVR models are not raw data points, but features that have been carefully engineered from the raw data based, in part, on insights gained from cases. Sometimes, simple data combinations suffice; for example, we could see during the case study that the difference between air and skin temperature was more relevant than either individual measurement. Other times, we have had to employ complex systems of equations; for example, a complex physiological model is needed to characterize the impact of insulin on BG levels. Even as we move toward more automated means of feature engineering and more reliance on machine learning techniques, early use of CBR can help to provide insight and intuition that may guide big data exploration and interpretation.

5 Summary and Conclusion

The 4DSS is a prototypical hybrid CBR system that aims to help T1D patients achieve and maintain good BG control. As cases and data have accumulated over ten years, the research emphasis has shifted toward using the accumulated data to build machine learning models for BG prediction. While these models have applicability to situation assessment within the 4DSS, their greater potential is in facilitating a wide range of practical applications that could enhance safety and quality of life for T1D patients. The recent proliferation of commercially available fitness bands has presented the opportunity to incorporate new types of data indicative of patient activity into the models to improve prediction accuracy. However, when it was initially unclear whether or how this new data might be leveraged, a case study was conducted, calling CBR back into play.

In the N-of-1 study, a T1D patient on insulin pump therapy with continuous glucose monitoring wore a fitness band and entered life-event data into the 4DSS database for two months. The aggregated data was displayed via custom-built visualization software and reviewed at weekly intervals by the patient, his physician, and AI researchers. BG control problems were analyzed with a focus on identifying patterns in the new data at the time the problems occurred. Some promising patterns could be visualized, including a marked rise in GSR with severe hypoglycemia. This case-based focus provided insight and intuition about how the new data relates to BG levels. Work on integrating the new data into our BG prediction models is currently underway.

6 Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1117489. Additional support was provided by Medtronic and Ohio University. We gratefully acknowledge the contributions of diabetologist Jay Shubrook, DO, graduate research assistants Kevin Plis and Samson Xia, and Research Experience for Undergraduates (REU) participants Hannah Quillin and Charlie Murphy.

References

1. Bichindaritz, I., Kansu, E., Sullivan, K.M.: Case-based reasoning in CAREPARTNER: Gathering evidence for evidence-based medical practice. In: Smyth, B., Cunningham, P. (eds.) Advances in Case-Based Reasoning: 4th European Workshop, Proceedings EWCBR-98. pp. 334–345. Springer-Verlag, Berlin (1998)
2. Bunescu, R., Struble, N., Marling, C., Shubrook, J., Schwartz, F.: Blood glucose level prediction using physiological models and support vector regression. In: Proceedings of the Twelfth International Conference on Machine Learning and Applications (ICMLA). pp. 135–140. IEEE Press (2013)
3. Cryer, P.E.: Hypoglycemia: Still the limiting factor in the glycemic management of diabetes. Endocrine Practice 14(6), 750–756 (2008)
4. Diabetes Control and Complications Trial Research Group: The effect of intensive treatment of diabetes on the development and progression of long-term complications in insulin-dependent diabetes mellitus. New England Journal of Medicine 329(14), 977–986 (1993)
5. IBM: IBM Watson Health (2015), http://www.ibm.com/smarterplanet/us/en/ibmwatson/health/, accessed July, 2015
6. Marling, C., Bunescu, R., Shubrook, J., Schwartz, F.: System overview: The 4 Diabetes Support System. In: Lamontagne, L., Recio-García, J.A. (eds.) Workshop Proceedings of the Twentieth International Conference on Case-Based Reasoning. pp. 81–86. Lyon, France (2012)
7. Marling, C., Wiley, M., Bunescu, R., Shubrook, J., Schwartz, F.: Emerging applications for intelligent diabetes management. AI Magazine 33(2), 67–78 (2012)
8. Plis, K., Bunescu, R., Marling, C., Shubrook, J., Schwartz, F.: A machine learning approach to predicting blood glucose levels for diabetes management. In: Modern Artificial Intelligence for Health Analytics: Papers Presented at the Twenty-Eighth AAAI Conference on Artificial Intelligence. pp. 35–39. AAAI Press (2014)
9. Schwartz, F.L., Shubrook, J.H., Marling, C.R.: Use of case-based reasoning to enhance intensive management of patients on insulin pump therapy. Journal of Diabetes Science and Technology 2(4), 603–611 (2008)
10. Tidepool: The Tidepool Platform: A home for diabetes data (2015), http://tidepool.org/platform/, accessed August, 2015
11. World Health Organization: 10 facts about diabetes (2014), http://www.who.int/features/factfiles/diabetes/facts/en/, accessed July, 2015

Data Mining Methods for Case-Based Reasoning in Health Sciences

Isabelle Bichindaritz
Computer Science Department, State University of New York
Oswego, NY, USA

Abstract. Case-based reasoning (CBR) systems often refer to diverse data mining functionalities and algorithms. This article locates examples, many from health sciences domains, mapping data mining functionalities to CBR tasks and steps, such as case mining, memory organization, case base reduction, generalized case mining, indexing, and weight mining. Data mining in CBR focuses greatly on incremental mining for memory structures and organization with the goal of improving performance of retrieval, reuse, revise, and retain steps. Researchers are aiming at the ideal memory as described in the theory of the dynamic memory, which follows a cognitive model, while also improving performance and accuracy in retrieve, reuse, revise, and retain steps. Several areas of potential cross-fertilization between CBR and data mining are also proposed.

1 Introduction

Case-based reasoning (CBR) systems have tight connections with machine learning and data mining as exemplified by their description in data mining (Han et al. 2012) and machine learning (Mitchell 1997) textbooks. They have been tagged by machine learning researchers as *lazy* learners because they defer the decision of how to generalize beyond the training set until a target new case is encountered (Mitchell 1997), by opposition to most other learners, tagged as *eager*. Even though a large part of the inductive inferences are definitely performed at *Retrieval* time in CBR (Aha 1997), mostly through sophisticated similarity evaluation, most CBR systems also perform inductive inferences at *Retain* time. There is a long tradition within this research community to study what is a memory, and what its components and organization should be. Indeed CBR methodology focuses more on the memory part of its intelligent systems (Schank 1982) than any other artificial intelligence (AI) methodology, and this often entails learning declarative memory structures and organization. This article proposes to review the main data mining functionalities and how they are used in CBR systems by describing examples of systems using them and analyzing which roles they play in the CBR framework (Aamodt and Plaza 1994). The research question addressed is to de-

termine the extent to which data mining functionalities are being used in CBR systems, to enlighten possible future research collaborations between these two fields, particularly in health sciences applications. This paper is organized as follows. After the introduction, the second section highlights major concepts and techniques in data mining. The third section reviews the main CBR cycle and principles. The fourth section explains relationships between CBR and machine learning. The following sections dive into several major data mining functionalities and how they relate to CBR. The ninth section summarizes the findings and proposes future directions. It is followed by the conclusion.

2 Data Mining Functionalities and Methods

Data mining is the analysis of observational data sets to find unsuspected relationships and to summarize the data in novel ways that are both understandable and useful to the data owner (Hand et al. 2001). Traditionally described as a misnomer, knowledge discovery or knowledge discovery in databases is a preferred term. Some functionalities are clearly well defined and researched, among which (Han et al. 2012):

- **Classification / prediction:** classification is a supervised data mining method applied to datasets containing an expert labeling in the form of a categorical attribute, called a class; when the attribute is numeric, the method is called prediction. Examples of classifiers include neural networks, support vector machines (SVMs), naïve Bayes, and decision trees.
- **Association Mining**: association mining mines for frequent itemsets in a dataset, which can be represented as rules such as in market basket analysis. It is an unsupervised method. The most famous algorithm in this category is a priori algorithm.
- **Clustering:** clustering finds groups of similar objects in a dataset, which are also dissimilar from the objects in other clusters. In addition to the similarity-based methods like K Means, some methods use density-based algorithms or hierarchical algorithms.

Considerations for evaluating the mining results vary in these different methods, however a set of quality measurements are traditionally associated with each, for example accuracy or error rate for classification, and lift or confidence for association mining.

These core functionalities can be combined and applied to several data types, with extensions to the underlying algorithms or completely new methods.in addition to the classical nominal and numeric data types. Well researched data types are graphs, texts, images, time series, networks, streams, etc. We refer to these extensions as multimedia mining.

Other types of functionalities, generally combined with the core ones are for example feature selection, where the goal is to select a subset of features, sampling, where the goal is to select a subset of input rows, and characterization,

where the goal is to provide a summary representation of a set of rows, for example those contained in a cluster.

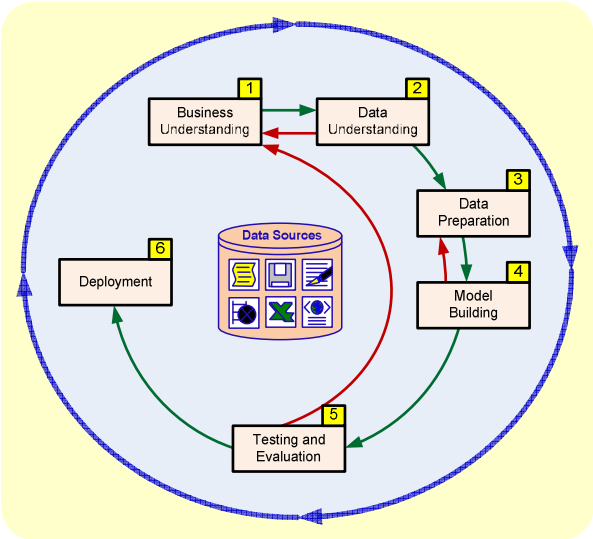


Fig. 1. CRISP-DM data mining process (Han et al. 2012)

Finally, the CRISP-DM methodology has been described to guide the data mining process (see Fig. 1) (Han et al. 2012). This methodology stresses the importance of stages preparing for and following the actual model building stage: data preparation, dealing with issues such as data consolidation, data cleaning, data transformation, and data reduction, which can require up to 85% of all the time dedicated to a project.

3 CBR Cycle and Methods

Case Based Reasoning is a problem solving methodology that aims at reusing previously solved and memorized problem situations, called cases. Traditionally, its reasoning cycle proceeds through steps (see Fig. 2). This article will refer to the major steps as Retrieve, Reuse, Revise, and Retain (Aamodt and Plaza 1994).

4 CBR and Machine Learning

CBR systems are generally classified as data mining systems because they can perform classification or prediction tasks. From a set of data – called cases in CBR – the classification or prediction achieved gives the case base a competency be-

yond what the data provide. If CBR systems are in par with data mining systems in such tasks as classification and prediction, there is, though an important difference. CBR systems start their reasoning from knowledge units, called cases, while data mining systems most often start from raw data. This is why case mining, which consists in mining raw data for these knowledge units called cases, is a data mining task often used in CBR. CBR systems also belong to instance based learning systems in the field of machine learning, defined as systems capable of automatically improving their performance over time. Although there is much commonality between data mining and machine learning, their definitions and goals are different. CBR systems are problem-solving systems following a reasoning cycle illustrated in Fig. 1. However as long as they learn new cases in their retain step, they are qualified as learning systems, thus belonging to machine learning system.

For this article, we will focus on identifying which data mining functionalities and methods are used in CBR, and what is their result in the CBR memory.

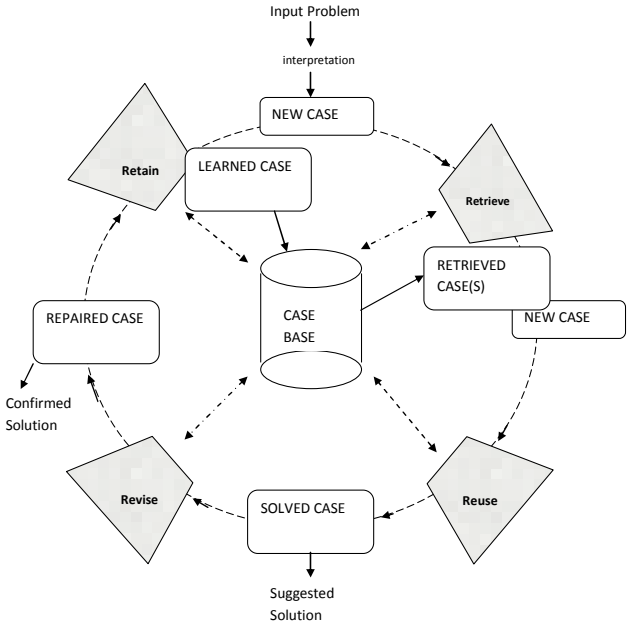


Fig 2. The classical CBR reasoning cycle (Aamodt and Plaza 1994)

First of all, since data mining emerged in the 90's from scaling up machine learning algorithms to large datasets, let us review what machine learning authors have been saying about CBR. They consider case-based reasoning systems as either analogical reasoning systems (Michalski 1993), or instance based learners (Mitchell 1997). Michalski (1993) presents the analogical inference, at the basis of case-based retrieval, as a dynamic induction performed during the matching process. Mitchell (1997) refers to CBR as a kind of instance based learner. This au-

thor labels these systems as *lazy* learners because they defer the decision about how to generalize beyond the training data until each new query instance is encountered. This allows CBR systems to not commit to a global approximation once and for all during the training phase of machine learning, but to generalize specifically for each target case, therefore to fit its approximation bias, or induction bias, to the case at hand. He points here to the drawback of overgeneralization that is well known for eager learners, from which instance based learners are exempt (Mitchell 1997).

These authors focus their analysis on the inferential aspects of learning in case-based reasoning. Historically CBR systems have evolved from the early work of Schank in the theory of the dynamic memory (Schank 1982), where this author proposes to design intelligent systems primarily by modeling their memory. Ever since Schank's precursory work on natural language understanding, one of the main goals of case-based reasoning has been to integrate as much as possible memory and inferences for the performance of intelligent tasks. Therefore focusing on studying how case-based reasoning systems learn, or mine, their memory structures and organization can prove at least as fruitful as studying and classifying them from an inference standpoint.

From a memory standpoint, learning in CBR consists in the creation and maintenance of the structures and organization in memory. It is often referred to as case base maintenance (Wilson and Leake 2001). In the general cycle of CBR, learning takes place within the reasoning cycle - see (Aamodt and Plaza 1994) for this classical cycle. It completely serves the reasoning, and therefore one of its characteristics is that it is an *incremental* type of mining. It is possible to fix it after a certain point, though; in certain types of applications, but it is not a tradition in CBR: *learning is an emergent behavior from normal functioning* (Kolodner 1993). When an external problem-solving source is available, CBR systems start reasoning from an empty memory, and their reasoning capabilities stem from their progressive learning from the cases they process. Aamodt and Plaza (1994) further state that *case-based reasoning favours learning from experience*. The decision to stop learning because the system is judged competent enough is not taken from definitive criteria. It is the consequence of individual decisions made about each case, to keep it or not in memory depending upon its potential contribution to the system. Thus often the decisions about each case, each structure in memory, allow the system to evolve progressively toward states as different as ongoing learning, in novice mode, and its termination, in expert mode. If reasoning and thus learning are directed from the memory, learning answers to a process of prediction of the conditions of cases recall (or retrieval). As the theory of the dynamic memory showed, recall and learning are closely linked (Schank 1982). Learning in case-based reasoning answers a disposition of the system to anticipate future situations: *the memory is directed toward the future* both to avoid situations having caused a problem and to reinforce the performance in success situations.

More precisely, learning in case-based reasoning, takes the following forms:

1. Adding a case to the memory: it is at the heart of CBR systems, traditionally one of the main phases in the reasoning cycle, and the last one: *Retain* (Aamodt

and Plaza 1994). It is the most primitive learning kind, also called learning by consolidation, or rote learning.
2. Explaining: the ability of a system to find explanations for its successes and failures, and by generalization the ability to anticipate.
3. Choosing the indices: it consists in anticipating *Retrieval*, the first reasoning step.
4. Learning memory structures: these may be learnt by generalization from cases or be provided from the start to hold the indices for example. These learnt memory structures can play additional roles, such as facilitating reuse or retrieval.
5. Organizing the memory: the memory comprises a network of cases, given memory structures, and learned memory structures, organized in efficient ways. Flat and hierarchical memories have been traditionally described.
6. Refining cases: cases may be updated, refined based upon the CBR result.
7. Discovering knowledge or metareasoning: the knowledge at the basis of the case-based reasoning can be refined, such as modifying the similarity measure (weight learning), or situation assessment refinement. For example d'Aquin et al. (2007) learn new adaptation rules through knowledge discovery.

5 Classification / Prediction and CBR

Since CBR is often used as a classifier, other classifiers are generally used in ensemble learning to combine the CBR expertise with other classification/prediction algorithms. Another type of combination of classifier is to use several CBR systems as input to another classifier, for example SVM, applied to the task of predicting business failure (Li and Sun 2009).

Another notable class of systems is composed of those performing *decision tree induction* to organize their memory. INRECA (Auriol et al. 1994) project studied how to integrate CBR and decision tree induction. They propose to preprocess the case base by an induction tree algorithm, namely a decision tree. Later refined into an *INRECA tree* (see Fig. 2), which is a hybrid between a decision tree and a k-d tree, this method allows both similarity based retrieval and decision tree retrieval, is incremental, and speeds up the retrieval. This system was used in biological domains among others.

6 Association Mining and CBR

Association mining, although not looking closely related to CBR, can be resorted in several scenarios. Main uses are for case mining and case base maintenance.

Wong et al. (2001) use fuzzy association rule mining to learn cases from a web log, for future reuse through CBR.

Liu et al. (2008) use frequent item sets mining to detect associations between cases, and thus detect cases candidate for removal from the case base and thus its reduction (Retain step).

7 Clustering and CBR

Memory structures in CBR are foremost cases. A case is defined as a contextualized piece of knowledge representing an experience that teaches a lesson fundamental to achieving the goals of a reasoner (Kolodner 1993). For many systems, cases are represented as truthfully as possible to the application domain. Additionally, data mining methods have been applied to cases themselves, features, and generalized cases. These techniques can be applied concurrently to the same problem, or selectively. If the trend is now to use them selectively, probably in the near future CBR systems will use these methods more and more concurrently.

7.1 Case mining

Case mining refers to the process of mining potentially large data sets for cases (Yang and Cheng 2003). Researchers have often noticed that cases simply do not exist in electronic format, that databases do not contain well-defined cases, and that the cases need to be created before CBR can be applied. Instead of starting CBR with an empty case base, when large databases are available, preprocessing these to learn cases for future CBR permits to capitalize on the experience dormant in these databases. Yang and Cheng (2003) propose to learn cases by linking several database tables through *clustering* and *Support Vector Machines (SVM)*. The approach can be applied to learning cases from electronic medical records (EMRs).

7.2 Generalized case mining

Generalized case mining refers to the process of mining databases for generalized and/or abstract cases. Generalized cases are named in varied ways, such as prototypical cases, abstract cases, prototypes, stereotypes, templates, classes, ossified cases, categories, concepts, and scripts – to name the main ones (Maximini et al. 2003). Although all these terms refer to slightly different concepts, they represent structures that have been abstracted or generalized from real cases either by the CBR system, or by an expert. When these prototypical cases are provided by a domain expert, this is a knowledge acquisition task. More frequently they are learnt from actual cases. In CBR, prototypical cases are often learnt to structure the memory. Therefore most of the prototypical cases presented here will also be listed in the section on structured memories.

In medical domains, many authors mine for *prototypes*, and simply refer to *induction* for learning these. CHROMA (Armengol and Plaza 1994) uses induction to learn prototypes corresponding to general cases. Bellazzi et al. organize their memory around prototypes (Bellazzi et al. 1998). The prototypes can either have been acquired from an expert, or induced from a large case base. Schmidt and Gierl (1998) point that prototypes are an essential knowledge structure to fill the gap between general knowledge and cases in medical domains. The main purpose of this prototype learning step is to guide the retrieval process and to decrease the amount of storage by erasing redundant cases. A generalization step becomes necessary to learn the knowledge contained in stored cases.

Others specifically refer to *generalization,* so that their prototypes correspond to generalized cases. For example Malek proposes to use a *neural network* to learn the prototypes in memory for a classification task, such as diagnosis (Malek 1995). Portinale and Torasso (1995) in ADAPTER organize their memory through E-MOPs (Kolodner 1993) learnt by generalization from cases for diagnostic problem-solving. Maximini et al. (2003) have studied the different structures induced from cases and point out that several different terms exist, such as generalized case, prototype, schema, script, and abstract case. The same terms do not always correspond to the same type of entity. They define three types of cases. A point case is what we refer to as a real or ground case. The values of all its attributes are known. A generalized case is an arbitrary subset of the attribute space.

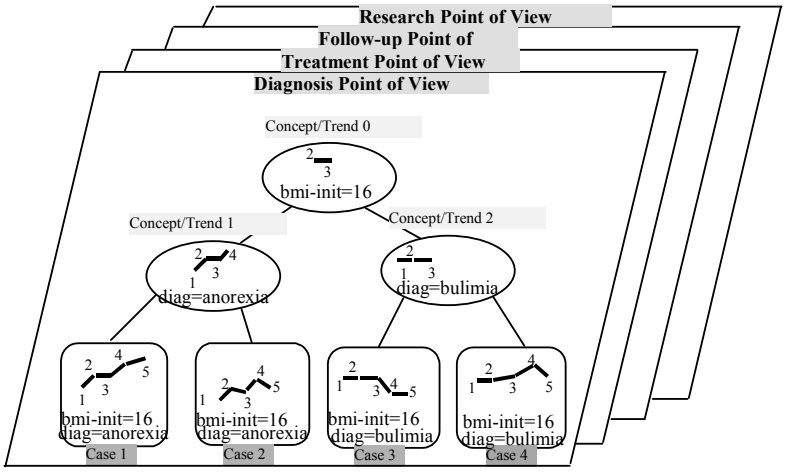


Fig. 3. Hierarchical memory organization in MNAOMIA: concepts are learnt during CBR for diagnosis, treatment, and/or follow-up, and can be reused by research task (Bichindaritz 1995)

There are two forms: the attribute independent generalized case, in which some attributes have been generalized (interval of values) or are unknown, and the attribute dependent generalized case, which cannot be defined from independent subsets of their attributes.

Finally, many authors learn *concepts* through *conceptual clustering*. MNAOMIA (Bichindaritz 1995) learns concepts and trends from cases through *conceptual clustering* (see Fig. 3). Perner learns a hierarchy of classes by *hierarchical conceptual clustering*, where the concepts represent clusters of prototypes (Perner 1998).

Dìaz-Agudo and Gonzàlez-Calero (2003) use *formal concept analysis* (FCA) – a mathematical method from data analysis - as another induction method for extracting knowledge from case bases, in the form of *concepts*. The authors point to one notable advantage of this method, during adaptation. The FCA structure induces dependencies among the attributes that guide the adaptation process (Dìaz-Agudo et al. 2003). Napoli (2010) stresses the important role FCA can play for classification purposes in CBR, through learning a case hierarchy, indexing, and information retrieval.

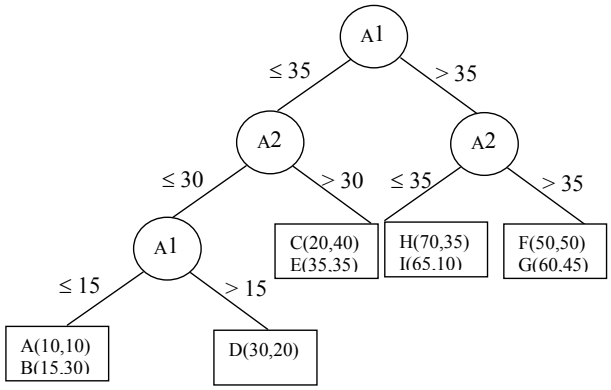


Fig. 4. Tree memory organization in INRECA using k-d trees (Auriol et al. 1994)

7.3 Mining for Memory Organization

Efficiency at case retrieval time is conditioned by a judicious memory organization. Two main classes of memory are presented here: unstructured – or flat – memories, and structured memories.

Flat memories

Flat memories are memories in which all cases are organized at the same level. Retrieval in such memories processes all the cases in memory. Classical nearest neighbor (kNN) retrieval is a method of choice for retrieval in flat memories. Flat memories can also contain prototypes, but in this case the prototypical cases do not serve as indexing structures for the cases. They can simply replace a cluster of similar cases that has been deleted from the case base during case base maintenance activity. They can also have been acquired from experts. Flat memories are

the memories of predilection of kNN retrieval methods (Aha 1997) and of so-called memory-based systems.

Structured memories

Among the different structured organizations, the accumulation of generalizations or abstractions facilitates the evaluation of the situation the control of indexation.

Structured memories, dynamic, present the advantage of being declarative. The important learning efforts in declarative learning are materialized in the structures and the dynamic organization of their memories. In medical imaging, Perner learns a hierarchy of classes by *hierarchical conceptual clustering*, where the concepts are clusters of prototypes (Perner 1998). She notes the advantages of this method: a more compact case base, and more robust (error-tolerant).

MNAOMIA (Bichindaritz 1995) proposes to use *incremental concept learning*, which is a form of hierarchical clustering, to organize the memory. This system integrates highly data mining with CBR because it reuses the learnt structures to answer higher level tasks such as generating hypotheses for clinical research (see Fig. 3), as a side effect of CBR for clinical diagnosis and treatment decision support. Therefore this system illustrates that by learning memory structures in the form of concepts, the classical CBR classification task improves, and at the same time the system extracts what it has learnt, thus adding a knowledge discovery dimension to the classification tasks performed.

Another important method, presented in CHROMA (Armengol and Plaza 1994), is to organize the memory like a hierarchy of objects, by *subsomption*. Retrieval is then a classification in a hierarchy of objects, and functions by substitution of values in slots. CHROMA uses its prototypes, induced from cases, to organize its memory. The retrieval step of CBR retrieves relevant prototypes by using subsomption in the object oriented language NOOS to find the matching prototypes.

Many systems use personalized memory organizations structured around several layers or *networks*, for example neural networks (Malek 1995).

Another type of memory organization is the *formal concept lattice*. Dìaz-Agudo and Gonzàlez-Calero (2003) organize through formal concept analysis (FCA) the case base around *Galois lattices*. Retrieval step is a classification in a concept hierarchy, as specified in the FCA methodology, which provides such algorithms (Napoli 2010). The concepts can be seen as an alternate form of indexing structure.

Yet other authors take advantage of the *B-tree structure* implementing databases and retrieve cases using database SQL query language over a large case base stored in a database (West and McDonald 2003).

8 Feature Selection and CBR

Feature mining refers to the process of mining data sets for features. Many CBR systems select the features for their cases, and/or generalize them. Wiratunga et al.

(2004) notice that transforming textual documents into cases requires dimension reduction and/or feature selection, and show that this preprocessing improves the classification in terms of CBR accuracy – and efficiency. These authors induce a kind of decision tree called boosted *decision stumps,* comprised of only one level, in order to select features, and *induce rules* to generalize the features. In biomedical domains, in particular when data vary continuously, the need to abstract features from streams of data is particularly prevalent. Other, and notable, examples include Montani et al., who reduce their cases time series dimensions through *Discrete Fourier Transform* (Montani et al. 2004), approach adopted by other authors for time series (Nilsson and Funk 2004). Niloofar and Jurisica propose an original method for generalizing features. Here the generalization is an abstraction that reduces the number of features stored in a case (Niloofar and Jurisica 2004). Applied to the bioinformatics domain of micro arrays, the system uses both *clustering* techniques to group the cases into clusters containing similar cases, and *feature selection* techniques.

Table 1. Data mining functionalities versus CBR steps map – methods italicized represent future directions

	Classification / prediction	***Association mining***	***Clustering***	***Feature selection***
Data preparation / Metareasoning	**Ensemble learning**	**Case mining**	**Case mining**	
Retrieve	***Opportunistic similarity knowledge mining***			
Reuse	***Opportunistic reuse knowledge mining***			
Revise	***Opportunistic revise knowledge mining***			
Retain	**Memory organization**	**Case base reduction**	**Generalized case mining** **Memory organization**	**Indexing** **Weight learning**

9 Discussion and Future Directions

In addition to the main functionalities listed above, multimedia mining extends the algorithms to the form taken by cases and the type of their features for the same kinds of applications previously listed.

In summary, if we map the different data mining functionalities and the CBR steps / tasks, we notice on Table 1 that the steps benefitting the most from data mining are Retain, Data preparation and Metareasoning. This is not surprising because these steps are the most involved in declarative knowledge learning or updating. However the processing intensive steps such as Retrieve, Reuse and Re-

vise do not seem to resort to data mining beside the dynamic induction mentioned in Section 4.

Interesting areas to explore could be feature selection functionality for case mining, data preparation, or metareasoning. Retrieve, Reuse, and Revise could also explore the use of data mining. For retrieval, in addition to weight learning already mentioned, learning similarity measures (Stahl 2005), or improving on an existing one, would be valuable. For reuse or revise, learning adaptation rules or revision rules or models would be highly pertinent – and some work has started in these areas (Badra et al. 2009). These synergies could take place during the Retain step, but also in an opportunistic fashion during the processing steps (see Table 1).

We can also foresee such synergies with Big Data for the processing of large datasets in distributed main memory that can make efficient use of data mining during processing on a larger scale. It is therefore very important for CBR researchers and professionals to gain expertise in data mining advances and their applicability to CBR.

CBR research focuses mostly on the model building stage of CRISP-DM. Other aspects of the CRISP-DM methodology would also be interesting for CBR synergies, for example aspects of data understanding, data preparation, testing, evaluation, and deployment in relationship with CBR to make this methodology more robust to fielded applications.

10 Conclusion

CBR systems make efficient use of most data mining tasks defined for descriptive modeling. We can list among the main ones encountered in biomedical domains, cluster analysis, rule induction, hierarchical cluster analysis, and decision tree induction. The motivations for performing an incremental type of data mining during CBR are several folds, and their efficiency has been measured to validate the approach. The main motivations are the following:

- Increase efficiency of retrieval mostly, but also of reuse, revise, and retain steps.
- Increase robustness, tolerance to noise.
- Increase reasoning accuracy and effectiveness.
- Improve storage needs.
- Follow a cognitive model.
- Add functionality, such as a synthetic task like generating new research hypotheses as a side effect of normal CBR functioning.
- Perform metareasoning, such as knowledge discovery to learn new adaptation rules.

The memory organization maps directly into the retrieval method used. For example, generalized cases and the like are used both as indexing structures, and organizational structures. We can see here a direct mapping with the theory of the

dynamic memory, which constantly influences the CBR approach. The general idea is that the learned memory structures and organizations condition what inferences will be performed, and how. This is a major difference with database approaches, which concentrate only on retrieval, and also with data mining approaches, which concentrate only on the structures learned, and not on how they will be used. Opportunistic use of data mining during the retrieval, reuse, and revise steps would bring a more robust dimension to CBR by learning when a need arises, instead of, or in addition to, systematically at Retain. The ideal CBR memory is one which at the same time speeds up the retrieval step, and improves effectiveness, efficiency, and robustness of the task performed by the reasoner, and particularly the reuse performed, influencing positively both the retrieval, the reuse and the other steps. Researchers do not want to settle for a faster retrieval at the expense of less accuracy due to an overgeneralization. And they succeed at it.

Future work involves revisiting these data mining techniques in the framework of the knowledge containers identified by Richter (2003) and constantly tracking novel methods used as they appear. The variety of approaches as well as the specific and complex purpose lead to thinking that there is space for future models and theories of CBR memories, in particular embracing metareasoning and opportunistic approaches more systematically, and where data mining will play a larger role.

11 References

Aamodt A, Plaza E (1994) Case-Based Reasoning: Foundational Issues, Methodologies Variations, and Systems Approaches. AI Communications, IOS Press, Vol. 7: 1:39-59

Aha DW (1997) Lazy Learning. Artificial Intelligence Review 11:7-10

Armengol E, Plaza E (1994) Integrating induction in a case-based reasoner. In: Keane M, Haton JP, Manago M (eds) Proceedings of EWCBR 94. Acknosoft Press, Paris, pp 243-251

Auriol E, Manago M, Althoff KD, Wess S, Dittrich S (1994) Integrating Induction and Case-Based Reasoning: Methodological Approach and First Evaluations. In: Keane M, Haton JP, Manago M (eds) Proceedings of EWCBR 94. Acknosoft Press, Paris, pp 145-155

Badra F, Cordier A, Lieber J (2009) Opportunistic Adaptation Knowledge Discovery. In: McGinty L, Wilson DC (eds) Proceedings of ICCBR 09. Springer-Verlag, Lecture Notes in Artificial Intelligence, Berlin, Heidelberg, New York, pp 60-74

Bellazzi R, Montani S, Portinale L (1998) Retrieval in a Prototype-Based Case Library: A Case Study in Diabetes Therapy Revision. In: Smyth B, Cunningham P (eds) Proceedings of ECCBR 98. Springer-Verlag, Lecture Notes in Artificial Intelligence 1488, Berlin, Heidelberg, New York, pp 64-75.

Bichindaritz I (1995) A case-based reasoner adaptive to several cognitive tasks. In: Veloso M., Aamodt A (eds) Proceedings of ICCBR 95. Springer-Verlag,

Lecture Notes in Artificial Intelligence 1010, Berlin, Heidelberg, New York, pp 391-400
d'Aquin M, Badra F, Lafrogne S, Lieber J, Napoli A, Szathmary L (2007) Case Base Mining for Adaptation Knowledge Acquisition. In : IJCAI. 7, pp. 750-755
Dìaz-Agudo B, Gervàz P, Gonzàlez-Calero P (2003) Adaptation Guided Retrieval Based on Formal Concept Analysis. In: Ashley K, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 131-145
Han J, Kamber M, Pei J (2012) Data Mining concepts and Techniques. Morgan Kaufmann, Waltham, Massachussetts
Hand D, Mannila H, Smyth P (2001) Principles of Data Mining. The MIT Press, Cambridge, Massachusetts
Kolodner JL (1993) Case-Based Reasoning. Morgan Kaufmann Publishers, San Mateo, California
Leake, DB, & Wilson, DC (1998). Categorizing Case-base Maintenance: Dimensions and Directions. In: Advances in Case-Based Reasoning. Springer Berlin Heidelberg, pp. 196-207
Li H, Sun J, (2009) Predicting business failure using multiple case-based reasoning combined with support vector machine, Expert Systems with Applications, Volume 36, Issue 6, pp 10085-10096
Liu C-H, Chen L-S, Hsu C-C (2008) An association-based case reduction technique for case-based reasoning, Information Sciences, Volume 178, Issue 17 pp. 3347-3355
Malek M (1995) A Connectionist Indexing Approach for CBR Systems. In: Veloso M, Aamodt A (eds) Proceedings of ICCBR 95. Springer-Verlag, Lecture Notes in Artificial Intelligence 1010, Berlin, Heidelberg, New York, pp 520-527
Maximini K, Maximini R, Bergmann R (2003) An Investigation of Generalized Cases. In: Ashley KD, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 261-275
Michalski RS (1993) Toward a Unified Theory of Learning. In: Buchanan BG, Wilkins DC (eds) Readings in knowledge acquisition and learning, automating the construction and improvement of expert systems. Morgan Kaufmann Publishers, San Mateo, California, pp 7-38
Mitchell TM (1997) Machine Learning. Mc Graw Hill, Boston, Massachusetts
Montani S, Portinale L, Bellazzi R, Leornardi G (2004) RHENE: A Case Retrieval System for Hemodialysis Cases with Dynamically Monitored Parameters. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 659-672
Napoli A (2010) Why and How Knowledge Discovery Can Be Useful for Solving Problems with CBR. In: Proceedings of ICCBR 10. Springer-Verlag, Lecture Notes in Artificial Intelligence, Berlin, Heidelberg, New York, pp. 12-19

Niloofar A, Jurisica I (2004) Maintaining Case-Based Reasoning Systems: A Machine Learning Approach. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 17-31

Nilsson M, Funk P (2004) A Case-Based Classification of Respiratory sinus Arrhythmia. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 673-685

Perner P (1998) Different Learning Strategies in a Case-Based Reasoning System for Image Interpretation. In: Smyth B, Cunningham P (eds) Proceedings of ECCBR 98. Springer-Verlag, Lecture Notes in Artificial Intelligence 1488, Berlin, Heidelberg, New York, pp 251-261

Portinale L, Torasso P (1995) ADAPTER: An Integrated Diagnostic System Combining Case-Based and Abductive Reasoning. In: Veloso M, Aamodt A (eds) Proceedings of ICCBR 95. Springer-Verlag, Lecture Notes in Artificial Intelligence 1010, Berlin, Heidelberg, New York, pp 277-288

Richter MM (2003). Knowledge containers. In: Readings in Case-Based Reasoning. Morgan Kaufmann Publishers

Schank RC (1982) Dynamic memory. A theory of reminding and learning in computers and people. Cambridge University Press, Cambridge

Schmidt R, Gierl L (1998) Experiences with Prototype Designs and Retrieval Methods in Medical Case-Based Reasoning Systems. In: Smyth B, Cunningham P (eds) Proceedings of ECCBR 98. Springer-Verlag, Lecture Notes in Artificial Intelligence 1488, Berlin, Heidelberg, New York, pp 370-381

Stahl A (2005) Learning Similarity Measures: A Formal View Based on a Generalized CBR Model. In: Munoz-Avila H, Ricci F (eds): Proceedings of ICCBR 05. Springer-Verlag, Lecture Notes in Artificial Intelligence 3620, Berlin, Heidelberg, New York, pp 507-521

West GM, McDonald JR (2003) An SQL-Based Approach to Similarity Assessment within a Relational Database. In: Ashley K, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 610-621

Wilson DC, Leake DB (2001) Maintaining Case-based Reasoners: Dimensions and Directions. Computational Intelligence Journal, Vo. 17, No. 2:196–213

Wiratunga N, Koychev I, Massie S (2004) Feature Selection and Generalisation for Retrieval of Textual Cases. In: Funk P, Gonzàlez Calero P (eds) Proceedings of ECCBR 04. Springer-Verlag, Lecture Notes in Artificial Intelligence 3155, Berlin, Heidelberg, New York, pp 806-820

Wong C, Shiu S, Pal S (2001) Mining fuzzy association rules for web access case adaptation. In *Workshop Proceedings of Soft Computing in Case-Based Reasoning Workshop, Vancouver, Canada*, pp. 213-220

Yang Q, Cheng H (2003) Case Mining from Large Databases. In: Ashley K, Bridge DG (eds) Proceedings of ICCBR 03. Springer-Verlag, Lecture Notes in Artificial Intelligence 2689, Berlin, Heidelberg, New York, pp 691-702

Why hybrid Case-Based Reasoning will change the future of health science and healthcare

Peter Funk

School of Innovation, Design and Engineering,
Mälardalen University, PO Box 883 SE-721 23, Västerås, Sweden
{firstname.lastname}@mdh.se

Abstract, The rapid development of the medical field makes it impossible even for experts in the field to keep up with new treatments. By ensuring confidentiality and collecting structured cases on a large scale will enable clinical decision support far beyond what is possible today and will be a major leap in healthcare.

20 years ago physicians met and discussed medical cases over a cup of coffee or in the cafeteria, an efficient way of sharing experience and disseminating knowledge. Times are changing, physicians say they don't have time for this any more. In a modern and efficient healthcare organisation there is no longer room for experience sharing and patients are treated according to guidelines. Many physicians I have discussed with admit that the consequence is that as much as 30% of patients don't receive optimal treatment. Research in medicine is progressing so fast that it takes years for new results to spread and even specialists in their field are not able to keep up to date with all what is happening.

Imagine you have a patient in front of you, the CBR system immediately says "your patient's symptoms are very similar to 47 other patients in Europe, where there are 4 different treatments, patients with treatment C recovered within 2 weeks, twice as fast as with treatment A and B, there is no difference in treatment cost. Based on my experience (all my cases) a modification of treatment C is recommend (due to your patient having diabetes). In France there is an alternative treatment D (8 patients) with recovery time of less than 10 days, the cost for this treatment is 5 times higher". The system offers

- Advice at the point of care
- Dissemination of experience from new treatments/procedures
- Second option for an experienced clinician, transfer experience to a less experienced clinicians
- It can explain and justify all its conclusions and findings

We can provide all this with CBR and I cannot see how this can be solved without case-based clinical decision support systems. All the different foundational methods and

techniques are already available in research, to mention some [1,2,3], but to achieve a transformation of the healthcare system we need a large scale approach since it requires a change in how patient cases are recorded and stored in order to preserve privacy and enable experience reuse.

To achieve this we need more elaborate case structures enabling hybrid case-based reasoning including experience sharing, knowledge discovery, data mining. One approach is to use a two-layered structure used in the Pain-Out project [3,4].

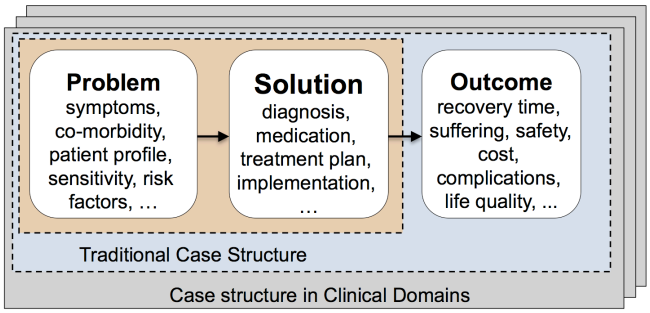


Fig 1. Extended case structure used in clinical application [3]

When explaining the concept of case-based reasoning for clinicians, the response by is often "such a tool would dramatically change and improve healthcare":

- Patient records become sources of experience and knowledge and provide supplementary information not currently accessible for diagnosis and treatment by clinicians at the point of care
- Clinicians will be able to easily and instantly share experience around specific case issues
- Dissemination of new clinical experience will be efficient and at the point of need.
- Patients will receive personalised and more informed diagnosis and care.

Reference

[1] Begum, Shahina, et al. "Case-based reasoning systems in the health sciences: a survey of recent trends and developments." *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 41.4 (2011): 421-434.

[2] C Marling, S Montani, I Bichindaritz, P Funk, Synergistic case-based reasoning in medical domains Expert systems with applications, 41. 2 (2014): 249-259.

[3] Ahmed M.U., Funk P., A Computer Aided System for Post-operative Pain Treatment Combining Knowledge Discovery and Case-Based Reasoning, In *Case-Based Reasoning Research and Development*, pp. 3-16. Springer Berlin Heidelberg, 2012.

[4] Rothaug et. al, Patients' perception of postoperative pain management: Validation of the International Pain Outcomes (IPO) Questionnaire, The Journal of Pain 14.11 (2013): 1361-1370, Churchill Livingstone.

Computer Cooking Contest

Workshop at the
Twenty-Third International Conference on
Case-Based Reasoning
(ICCBR 2015)

Frankfurt, Germany
September 2015

Emmanuel Nauer and David C. Wilson (Editors)

Program Chairs

Emmanuel Nauer	Loria, Université de Lorraine, France
David C. Wilson	University of North Carolina at Charlotte, USA

Technical Chair

Emmanuelle Gaillard	Loria, Université de Lorraine, France

Program Committee

David Aha	Naval Research Laboratory, USA
Klaus-Dieter Althoff	University of Hildesheim, Germany
Ralph Bergmann	University of Trier, Germany
Isabelle Bichindaritz	State University of New York at Oswego, USA
Ichiro Ide	Nagoya University, Japan
Luc Lamontagne	Université Laval, Canada
David Leake	Indiana University, USA
Jean Lieber	LORIA, Université de Lorraine, France
Mirjam Minor	Goethe University, Germany
Santiago Ontañón	Drexel University, USA
Miltos Petridis	Brighton University, UK
Ralph Traphöner	empolis GmbH, Germany
Nicolas Valance	Groupe SEB, France

Preface

We are happy to present the contributions of four teams that have been accepted to the Computer Cooking Contest 2015. The Computer Cooking Contest (CCC) is an open competition. All individuals (e.g., students, professionals), research groups, and others are invited to submit software that creates recipes. The primary knowledge source is a database of basic recipes from which appropriate recipes can be selected, modified, or even combined. The queries to the system will include the desired and undesired ingredients. For most of the queries there is no single correct or best answer. That is, many different solutions are possible, depending on the creativity of the software. There is no restriction on the technology that may be used; all are welcome. The only restriction is that the given database of recipes must be used as a starting point.

The 8th Computer Cooking Contest will be held in conjunction with the 2015 International Conference on Case-Based Reasoning in Frankfurt, Germany. A web site with detailed information is online at: computercookingcontest.net. There are three challenges:

1. **Cocktail challenge on making real cocktails**
   In this challenge, the system should be able to suggest a tasty cocktail recipe that matches a user query including a set of desired ingredients from a limited set of available ingredients and avoiding unwanted ones (not necessarily from the limited set of ingredients). In addition, the system should adapt the ingredient quantities. Without information on the ingredient quantities, the original quantities will be used to prepare the cocktail.
   - *Evaluation criteria*: scientific quality, culinary quality
   - *Assessment procedure*: paper evaluation and comparison of the results of the systems on a same set of queries by the jury (cocktail jury prize) and public vote after tasting in real the recipes of all the system on a same query chosen by the jury (cocktail public prize).
   - *Material provided by the organizers (see resources section)*: the cocktail case base of 109 cocktail recipes, semantically annotated according to the Wikitaaable ontology; access to the WikiTaaable ontology (wikitaaable.loria.fr); specified basic set of ingredients available to prepare a cocktail.
2. **Sandwich challenge on making real sandwiches**
   In this challenge, the system should be able to suggest a tasty cold sandwich recipe that matches a user query including a set of desired ingredients and avoiding unwanted ones. In addition, the system should adapt the recipe preparation, at least the order in which ingredient will be put in the sandwich. Without information on the ingredient quantities, the original input procedure will be used to prepare the sandwich in real. The recipe will be interpreted by a chef, to correct some usual missing preparation step, for example, put the tomatoes without mentioning that the tomatoes must be sliced.
   - *Evaluation criteria*: scientific quality, culinary quality

- *Assessment procedure*: paper evaluation and comparison of the results of the systems on a same set of queries by the jury (sandwich jury prize), and public vote after tasting in real the recipes of all the system on a same query chosen by the jury (sandwich public prize).
- *Material provided by the organizers*: the sandwich case base - a set of 21 sandwich recipes, semantically annotated according to the Wikitaaable ontology; access to the WikiTaaable ontology (wikitaaable.loria.fr); an additional database containing 9507 sandwich recipes crawled from the web.

3. **Open challenge on adapting cooking recipes**
   In this challenge, you may propose whatever you want about the adaptation of cooking recipes, e.g. workflow adaptation, text adaptation, community-based adaptation, recipes combination, explanations, similarity computation, recipe personalized recommendation, etc. The evaluation will take into account the originality aspect and the scientific aspect of the work. A running system implementing the work is optional.
   - *Evaluation criteria*: scientific quality, originality, culinary quality
   - *Assessment procedure*: usual scientific review process; program committee vote

We would like to thank all contributors, reviewers, local organizers, the jury, and our sponsors: Empolis, INRIA Nancy Grand Est, and LORIA, who kindly provided financial support for the CCC. We are looking forward to try some excellent sandwiches and fascinating, novel cocktails in Frankfurt.

September 2015 — Emmanuel Nauer
Frankfurt — David C. Wilson

Improving Ingredient Substitution using Formal Concept Analysis and Adaptation of Ingredient Quantities with Mixed Linear Optimization

Emmanuelle Gaillard, Jean Lieber, and Emmanuel Nauer

Université de Lorraine, LORIA — 54506 Vandœuvre-lès-Nancy, France
CNRS — 54506 Vandœuvre-lès-Nancy, France
Inria — 54602 Villers-lès-Nancy, France
firstname.lastname@loria.fr

Abstract. This paper presents the participation of the TAAABLE team to the 2015 Computer Cooking Contest. The TAAABLE system addresses the *mixology* and the *sandwich* challenges. For the *mixology challenge*, the 2014 TAAABLE system was extended in two ways. First, a formal concept analysis approach is used to improve the ingredient substitution, which must take into account a limited set of available foods. Second, the adaptation of the ingredient quantities has also been improved in order to be more realistic with a real cooking setting. The adaptation of the ingredient quantities is based on a mixed linear optimization. The team also applied TAAABLE to the *sandwich challenge*.

Keywords: case-based reasoning, formal concept analysis, adaptation of ingredient quantities, mixed linear optimization.

1 Introduction

This paper presents the participation of the TAAABLE team to the *mixology* and to the *sandwich* challenges of the 2015 Computer Cooking Contest (CCC). The TAAABLE system is based on many methods and techniques in the area of knowledge representation, knowledge management and natural language processing [1]. Currently, it is built over TUUURBINE (`http://tuuurbine.loria.fr`), a generic case-based reasoning (CBR) system over RDFS [2] which allows reasoning over knowledge stored in a RDF store, as the one provided by the contest.

For this edition of the CCC, TAAABLE has been extended in order to improve the ingredient substitution procedure which must manage unavailable foods. An approach based on formal concept analysis (FCA) allows improving ingredient substitutions. Moreover, the adaptation of the ingredient quantities has also been improved in order to be more realistic with a real cooking setting. The adaptation of the ingredient quantities is based on mixed linear optimization. This adaptation takes into account the preference unit given in the source recipe and proposes quantities which are usual. For example, when the ingredient is a lemon, its quantity will take the form of a human easy understandable value (i.e. a quarter, a half, etc. instead of *54 g*, which corresponds to a half lemon).

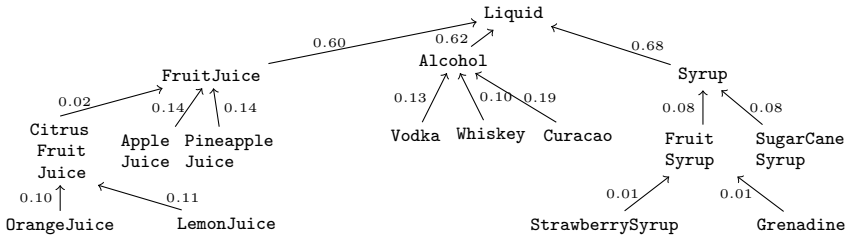


Fig. 1. The hierarchy forming the domain knowledge used in the running example with the generalization costs used as retrieval knowledge.

Section 2 introduces the core of the TAAABLE system. Section 3 details the new approaches developed specially for the mixology challenge. Section 4 explains the system submitted for the sandwich challenge.

2 The TAAABLE system

The challenges, proposed by the CCC since its first edition consists in proposing, according to a set of initial recipes, one or more recipes matching a user query composed of a set of wanted ingredients and a set of unwanted ingredients. The TAAABLE system addresses this issue through an instantiation of the generic CBR TUUURBINE system [3], which implements a generic CBR mechanism in which adaptation consists in retrieving similar cases and in replacing some features of these cases in order to adapt them as a solution to a query.

2.1 TUUURBINE founding principles

TUUURBINE is a generic CBR system over RDFS. The domain knowledge is represented by an RDFS base `DK` consisting of a set of triples of the form $\langle C \ \texttt{subClassOf} \ D\rangle$ where C and D are classes which belong to a same hierarchy (e.g, the food hierachy). Fig. 1 represents the domain knowledge for the running examples by a hierarchy whose edges $C \xrightarrow{x} D$ represent the triples $\langle C \ \texttt{subClassOf} \ D\rangle$. The retrieval knowledge is encoded by a cost function: $\texttt{cost}(\langle C \ \texttt{subClassOf} \ D\rangle) = x$ for an edge $C \xrightarrow{x} D$. This cost can be understood intuitively as the measure of "the generalization effort" from C to D. How this cost is computed is detailed in [1].

A TUUURBINE case `case` is described by a set of triples of the form $\langle URI_{\texttt{case}} \ \texttt{prop} \ \texttt{val}\rangle$, where $URI_{\texttt{case}}$ is the URI of `case`, `val` is either a resource representing a class of the ontology or a value and `prop` is an RDF property linking `case` to a hierarchy class or to the value. For simplification, in this paper, we represent a case by a conjunction of expressions only of the form `prop` : `val`. For example, the "Rainbow" recipe is represented by the following index `R`, which means that "Rainbow" is a cocktail recipe made from vodka, orange juice, grenadine and curacao (`ing` stands for *ingredient*).

$$\begin{aligned}\texttt{R} = \ & \texttt{dishType}:\texttt{CocktailDish} \\ & \wedge \texttt{ing}:\texttt{Vodka} \wedge \texttt{ing}:\texttt{OrangeJuice} \wedge \texttt{ing}:\texttt{Grenadine} \wedge \texttt{ing}:\texttt{Curacao}\end{aligned} \tag{1}$$

For instance, the first conjunct of this expression means that the triple $\langle URI_R$ `dishType CocktailDish`$\rangle$ belongs to the knowledge base.

2.2 TUUURBINE query

A TUUURBINE query is a conjunction of expressions of the form `sign prop : val` where `sign` $\in \{\epsilon, +, !, -\}$, `val` is a resource representing a class of the ontology and `prop` is an RDF property belonging to the set of properties used to represent cases. For example,

$$\begin{aligned} \texttt{Q} = {} & +\texttt{dishType} : \texttt{CocktailDish} \\ & \wedge \texttt{ing} : \texttt{Vodka} \wedge \texttt{ing} : \texttt{Grenadine} \wedge !\texttt{ing} : \texttt{Whiskey} \end{aligned} \tag{2}$$

is a query to search "a cocktail with vodka and grenadine syrup but without whiskey".

The signs ϵ (*empty sign*) and + are "positive signs": they prefix features that the requested case must have. + indicates that this feature must also occur in the source case whereas ϵ indicates that the source case may not have this feature, thus the adaptation phase has to make it appear in the final case.

The signs ! and − are "negative signs": they prefix features that the requested case must not have. − indicates that this feature must not occur in the source case whereas ! indicates that the source case may have this feature, and, if so, that the adaptation phase has to remove it.

2.3 TUUURBINE retrieval process

The retrieval process consists in searching for cases that best match the query. If an exact match exists, the corresponding cases are returned. For the query `Q` given in (2), the "Rainbow" recipe is retrieved without adaptation. Otherwise, the query is relaxed using a generalization function composed of one-step generalizations, which transforms `Q` (with a minimal cost) until at least one recipe of the case base matches $\Gamma(Q)$.

A one step-generalization is denoted by $\gamma = \texttt{prop} : \texttt{A} \rightsquigarrow \texttt{prop} : \texttt{B}$, where `A` and `B` are classes belonging to the same hierarchy with $\texttt{A} \sqsubseteq \texttt{B}$, and `prop` is a property used in the case definition. This one step-generalization can be applied only if `A` is prefixed by ϵ or ! in `Q`. If `A` is prefixed by !, thus `B` is necessarily the top class of the hierarchy. For example, the generalization of $!\texttt{ing} : \texttt{Rum}$ is $\epsilon\texttt{ing} : \texttt{Food}$, meaning that if rum is not wanted, it has to be replaced by some other food. Classes of the query prefixed by + and − cannot be generalized.

Each one-step generalization is associated with a cost denoted by $\texttt{cost}(A \rightsquigarrow B)$. The generalization Γ of `Q` is a composition of one-step generalizations $\gamma_1, \ldots \gamma_n$: $\Gamma = \gamma_n \circ \ldots \circ \gamma_1$, with $\texttt{cost}(\Gamma) = \sum_{i=1}^{n} \texttt{cost}(\gamma_i)$. For example, for:

$$\begin{aligned} \texttt{Q} = {} & +\texttt{dishType} : \texttt{CocktailDish} \\ & \wedge \texttt{ing} : \texttt{Vodka} \wedge \texttt{ing} : \texttt{PineappleJuice} \wedge \texttt{ing} : \texttt{Grenadine} \wedge !\texttt{ing} : \texttt{Whiskey} \end{aligned} \tag{3}$$

`PineappleJuice` is relaxed to `FruitJuice` according to the domain knowledge of Fig. 1. At this first step of generalization, $\Gamma(\texttt{Q}) =$

dishType : CocktailDish⊓ing : Vodka⊓ing : FruitJuice⊓!ing : Whiskey, which matches the recipe described in (1), indexed by OrangeJuice, a FruitJuice.

2.4 TUUURBINE adaptation process

When the initial query does not match existing cases, the cases retrieved after generalization have to be adapted. The adaptation consists of a specialization of the generalized query produced by the retrieval step. According to Γ(Q), to R, and to DK, the ingredient OrangeJuice is replaced with the ingredient PineappleJuice in R because FruitJuice of Γ(Q) subsumes both OrangeJuice and PineappleJuice. TUUURBINE implements also an adaptation based on rules where some ingredients are replaced with others in a given context [4]. For example, in cocktail recipes, replacing OrangeJuice and StrawberrySyrup with PineappleJuice and Grenadine is an example of an adaptation rule. This rule-based adaptation is directly integrated in the retrieval process by searching cases indexed by the substituted ingredients for a query about the replacing ingredients, for example by searching recipes containing OrangeJuice and StrawberrySyrup for a query about PineappleJuice and Grenadine.

2.5 TAAABLE as a TUUURBINE instantiation

The TAAABLE knowledge base is WIKITAAABLE (http://wikitaaable.loria.fr/), the knowledge base made available for this CCC edition. WIKITAAABLE is composed of the four classical knowledge containers: (1) the domain knowledge contains an ontology of the cooking domain which includes several hierarchies (about food, dish types, etc.), (2) the case base contains recipes described by their titles, the dish type they produce, the ingredients that are required, the preparation steps, etc., (3) the adaptation knowledge takes the form of adaptation rules as introduced previously, and (4) the retrieval knowledge, which is stored as cost values on subclass-of relations and adaptation rules.

In WIKITAAABLE, all the knowledge (cases, domain knowledge, costs, adaptation rules) is encoded in a triple store, because WIKITAAABLE uses Semantic Media Wiki, where semantic data is stored into a triple store. So, plugging TUUURBINE over the WIKITAAABLE triple store is quite easy because it requires only to configure TUUURBINE by giving the case base root URI, the ontology root URI and the set of properties on which reasoning may be applied.

3 Mixology challenge

The mixology challenge consists in retrieving a cocktail that matches a user query according to a set of available foods given by the CCC organizers (white rum, whiskey, vodka, orange juice, pineapple juice, sparkling water, coca-cola, beer grenadine syrup, lemon juice, mint leaves, lime, ice cube, brown sugar, salt, and pepper). TUUURBINE queries can express this kind of request using the ϵ and ! prefixes. Section 3.1 explains how the user query is transformed to take into account only the available foods, before being submitted to TUUURBINE. Two additionnal processes have been implemented to improve the TUUURBINE adaptation result. The first process searches, when some ingredients of the source

recipe are not available, the best way to replace them, or in some cases, to remove them (see Section 3.2). The second process uses REVISOR/CLC (see Section 3.4) to adapt quantities. A new formalization of the quantity adaptation problem is proposed to obtain more realistic quantity values, taking into account the type of unit given in the source case (see Section 3.4).

3.1 Query building

For the mixology challenge, where an answer must only contain the available food, the query may be built by adding to the initial user query the minimal set of classes of the food hierarchy that subsume the set of foods which are not available, each class being negatively prefixed by !. For example, let us assume that `OrangeJuice` and `PineappleJuice` are the only available fruit juices, that `Vodka` and `Whiskey` are the only available alcohols, that `SugarCaneSyrup` and `Grenadine` are the only available syrups, and that the user wants a cocktail recipe with `Vodka` but without `SugarCaneSyrup`. The initial user query will be $\texttt{Q} = +\texttt{dishType}:\texttt{CocktailDish} \wedge \epsilon\texttt{ing}:\texttt{Vodka} \wedge !\texttt{ing}:\texttt{SugarCane}$. According to Fig. 1, `LemonJuice`, `AppleJuice`, `Curacao`, and `StrawberrySyrup` will be added to this initial query with a ! for expressing that the result cannot contain one of these non available classes of food, which includes their descendant classes. The extended query `EQ` submitted to TUUURBINE will be:

$$\begin{aligned}\texttt{EQ} = \texttt{Q} &\wedge !\texttt{ing}:\texttt{LemonJuice} \wedge !\texttt{ing}:\texttt{AppleJuice} \\ &\wedge !\texttt{ing}:\texttt{StrawberrySyrup} \wedge !\texttt{ing}:\texttt{Curacao}\end{aligned}$$

For this example, TUUURBINE retrieves the "Rainbow" recipe with the adaptation "replace `Curacao` with `Food`", due to $!\texttt{ing}:\texttt{Curacao}$.

In order to replace `Curacao` by something more specific than `Food`, a new approach based on FCA is proposed.

3.2 Using FCA to search the best ingredient substitution

When ingredients of the source case must be replaced because these pieces of food are not available, we choose FCA to exploit ingredient combination in cocktail recipes in order to search which ingredient(s) is/are the most used with the ones already used in the recipe that must be adapted. FCA is a classification method allowing object grouping according to the properties they share [5]. FCA takes as input a *binary context*, i.e. a table in which objects are described by properties. Table 1 shows an example of binary context with 7 objects (which are cocktails), described by two kinds of properties: the ingredients they use, and some more generic ingredient classes: `_Alcohol`, the generic class of recipes with at least one alcohol, and `_Sugar`, the generic class of recipes with at least one sweet ingredient, like sugar or syrup. These generic classes are prefixed by `_` to be distinguished from the concrete ingredients. For example, the object `Screwdriver` has the properties `Vodka` and `Orange juice` (the ingredients used in this cocktail), and `_Alcohol`, because `Vodka` is an alcohol.

FCA produces *formal concepts* as output. A formal concept is a pair (I, E) where I is a set of properties, E is a set of objects, respectively called the *intent*

	_Alcohol	Vodka	White rum	Tequila	Cacha ca	Blue cucacao	Orange juice	Coca-cola	Lime	_Sugar	White sugar	Cane sugar syrup	Grenadine
Screwdriver	×	×					×						
Rainbow	×	×				×	×			×			×
Tequila sunrise	×			×			×			×			×
Ti'Punch	×		×						×	×		×	
Daiquiri	×		×						×	×		×	
Caipirinha	×				×				×	×	×		
Cuba libre	×		×					×	×				

Table 1. A binary context for cocktails, described by their ingredients and two generic food classes (`_Alcohol` and `_Sugar`).

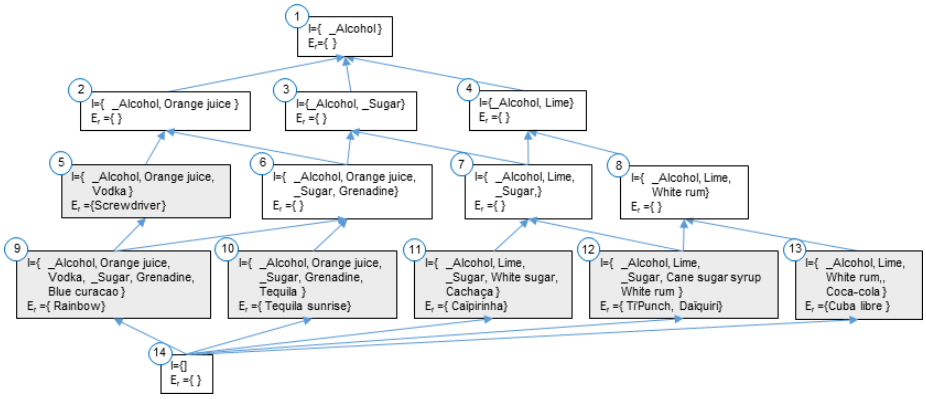


Fig. 2. Concept lattice organizing cocktails according to their ingredients.

and the *extent* of the formal concept, such that (1) I is the set of all properties shared by the objects of E and (2) E is the set of all objects sharing properties in I. The formal concepts can be ordered by extent inclusion, also called *specialisation* between concepts, into what is called a *concept lattice*. Fig. 2 illustrates the lattice corresponding to the binary context given in Table 1. On this figure, the extents E are given through a reduced form (noted E_r): the objects appear in the most specific concepts, the complete extent can be computed by the union of objects belonging to the subconcepts. So, the top concept (#1, in the figure) contains all the objects. In our example, its intent is `_Alcohol`, a property shared by all the objects. By contrast, the bottom concept is defined by the set of all properties. In our example, its extent is empty as none of the objects are described by all the properties.

To search a replacing ingredient in a given recipe or in a recipe according to pieces of food that will be kept, the idea is to exploit the lattice which captures concept similarities and organization. For example, concept #7, which intent is {`_Alcohol`, `Lime`, `_Sugar`}, allows an access to 3 cocktails containing at least one alcohol, at least one sugar, and lime. Adapting a cocktail can be based on the closeness between concepts. For example, when a replacing ingredient is searched

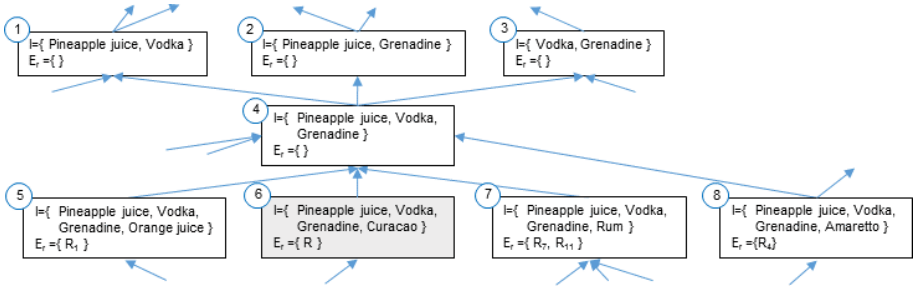


Fig. 3. Part of the concept lattice built from recipes using `PineappleJuice`, `Vodka`, and `Grenadine` (the ingredients that will be used in the resulting cocktail).

for *Cachaça* in the *Caipirinha* cocktail (in the intent of concept #11), some similar concepts (i.e. sharing a same super-concept) can be used. In the lattice given in example, concept #11 can be generalized to concept number #7, which extent contains cocktails with some alcohol, lime and some sugar. The cocktails in the extent of concept #12 are similar to the one of concept #11, because they share the `_Alcohol`, `Lime`, and `_Sugar` properties. When removing *"Cachaça"* from the *Caipirinha*, a possible ingredient for substitution, given by the lattice, could be `White rum`.

The approach exploiting the link between the concepts is used in many works using FCA for information retrieval. In Carpineto and Romano [6], the documents which are good answers to a query are searched in the lattice built from the document properties and from the query, around the concept representing the query. The same authors use this neighbour relation between concepts in a lattice for ordering documents returned by an information retrieval system [7].

Let C_R be the formal concept such that $E_r(C_R) = \{R\}$. A formal concept C close to C_R is searched according the following procedure. C is such that its intent $I(C)$ does not contain the substituting ingredient (`Curacao` in the example) and maximizes $|E_r(C)|$. First, C is searched in the ascendants of C_R, then in its siblings, and finally in the descendants of the siblings. The ingredient to be substituted is replaced by $I(C) \setminus I(C_R)$.

3.3 Real example of food substitution using FCA

To implement our approach, data about ingredient combinations in cocktail recipes has been collected. For this, we queried *Yummly* (`http://www.yummly.com/`). 16 queries were submitted; each query was composed of one ingredient (one available food) and was parametered to return all the *Yummly* cocktails and beverage recipes containing this ingredient. 9791 recipes have been collected. Unfortunately, the *Yummly* search engine does not necessarily return answers satisfying the query. So, the results are filtered, only to keep recipes that use at least one available food. Afterwards, the remaining recipes are deduplicated. After filtering and deduplicating, 6114 recipes are available, but only 1327 of them combine at least 2 available foods.

We show now, with query (3), how, after proposing to replace `OrangeJuice` with `PineappleJuice` and `StrawberrySyrup` with `Grenadine` in `R`, TAAABLE searches to replace `Curacao` which is not in the set of available foods. A part of the lattice resulting from the binary table containing recipes with `PineappleJuice`, `Grenadine` and `Vodka` is given in Fig. 3. Concept #6 corresponds to `R`, the recipe that must be adapted, and which has been added in the binary table to appear in the lattice. The most similar ingredient combination which includes `PineappleJuice`, `Grenadine` and `Vodka` is given by concept #7. Indeed, concept #8 cannot be used to produce a substitution because its intent contains `Amaretto` which is not an available food. Concept #5 intent contains `OrangeJuice`, an available food, but concept #5 is less close to concept #6 than concept #7, according to the selection procedure based on the maximal number of objects of E_r.

3.4 Adaptation of quantities with mixed integer linear optimization

Let us consider the following adaptation problem:

$\texttt{Source} =$

Recipe "Eggnog" (10 glasses)
10 cℓ of armagnac, 25 cℓ of rum, half a liter of milk, 5 eggs, 125 g of granulated sugar, 25 cℓ of fresh cream

$\texttt{Q}$ = "I want a cocktail recipe with cream but without egg or armagnac."

for which TUUURBINE produces the following ingredient substitution:

substitute egg and armagnac **with** banana and kirsch (4)

It must be noticed that this example does not comply with the constraints of the cocktail challenge (banana is not an available food), but has been chosen in order to illustrate various ideas related to adaptation of quantities. The approach to ingredient quantity adaptation is based on belief revision [8], applied to a formalization suited to adaptation of quantities. First, the adaptation problem $(\texttt{Source}, \texttt{Q})$ and the domain knowledge `DK` are formalized. Then, this adaptation process is described.

Formalization. Numerical variables are introduced to represent the ingredient quantities in a recipe. For the example, the following variables are introduced, for each food class `C`: $\texttt{alcohol}_\texttt{C}$, $\texttt{mass}_\texttt{C}$, $\texttt{number}_\texttt{C}$, $\texttt{sugar}_\texttt{C}$ and $\texttt{volume}_\texttt{C}$, which represent, respectively, the quantity (in grams) of alcohol in the ingredient `C` of the recipe, its mass (in grams), its number, its quantity (in grams) of sugar and its volume (in centiliters).[1] Therefore, the retrieved recipe can be expressed in this formalism by:

$$\begin{aligned}\texttt{Source} = {} & (\texttt{volume}_{\texttt{Armagnac}} = 10) \wedge (\texttt{volume}_{\texttt{Rum}} = 25) \wedge (\texttt{volume}_{\texttt{Milk}} = 50) \\ & \wedge (\texttt{number}_{\texttt{Egg}} = 5) \wedge (\texttt{mass}_{\texttt{GranulatedSugar}} = 125) \\ & \wedge (\texttt{volume}_{\texttt{FreshCream}} = 25)\end{aligned} \quad (5)$$

[1] One could consider other variables, e.g., the calories of ingredients, which would make possible to add constraints on the total number of calories in a dish.

In theory, all the variables could be continuous (represented by floating-point numbers). However, this can lead to adapted cases with, e.g., $\texttt{number}_{\texttt{Egg}} = 1.7$, which is avoided in most recipe books! For this reason, some variables v are declared as integer (denoted by $\tau(v) = \texttt{integer}$), the other ones as real numbers (denoted by $\tau(v) = \texttt{real}$).

The domain knowledge DK consists of a conjunction of *conversion equations*, *conservation equations* and *sign constraints*. The following conversion equations state that one egg without its shell has (on the average) a mass of 50 g, a volume of 5.2 cℓ, a quantity of sugar of 0.77 g and no alcohol:

$$\begin{array}{rr} \texttt{mass}_{\texttt{Egg}} = 50 \times \texttt{number}_{\texttt{Egg}} & \texttt{volume}_{\texttt{Egg}} = 5.2 \times \texttt{number}_{\texttt{Egg}} \\ \texttt{sugar}_{\texttt{Egg}} = 0.77 \times \texttt{number}_{\texttt{Egg}} & \texttt{alcohol}_{\texttt{Egg}} = 0. \end{array} \tag{6}$$

with $\tau(\texttt{mass}_{\texttt{Egg}}) = \tau(\texttt{volume}_{\texttt{Egg}}) = \tau(\texttt{sugar}_{\texttt{Egg}}) = \tau(\texttt{alcohol}_{\texttt{Egg}}) = \texttt{real}$ and $\tau(\texttt{number}_{\texttt{Egg}}) = \texttt{integer}$.

The following equations are also conjuncts of DK and represent the conservation of masses, volumes, etc.:

$$\begin{aligned} \texttt{mass}_{\texttt{EggOrEquivalent}} &= \texttt{mass}_{\texttt{Egg}} + \texttt{mass}_{\texttt{Banana}} \\ \texttt{volume}_{\texttt{Food}} &= \texttt{volume}_{\texttt{Liquid}} + \texttt{volume}_{\texttt{SolidFood}} \\ \texttt{volume}_{\texttt{Liquid}} &= \texttt{volume}_{\texttt{Brandy}} + \texttt{volume}_{\texttt{Rum}} + \texttt{volume}_{\texttt{FreshCream}} + \ldots \\ \texttt{volume}_{\texttt{Brandy}} &= \texttt{volume}_{\texttt{Armagnac}} + \texttt{volume}_{\texttt{Kirsch}} + \ldots \end{aligned} \tag{7}$$

where Food is the class of the food (any ingredient of a recipe is an instance of Food) and, e.g., $\texttt{alcohol}_{\texttt{Rum}}$ is related to $\texttt{volume}_{\texttt{Rum}}$ thanks to the conversion equation $\texttt{alcohol}_{\texttt{Rum}} = 0.4 \times \texttt{volume}_{\texttt{Rum}}$. Actually, equation (7) corresponds to the substitution of eggs by bananas.

Such conservation equations can be acquired using parts of the food hierarchy, thanks to some additional information. For instance, if C is a class of the hierarchy and $\{D_1, D_2, \ldots, D_p\}$ is a set of subclasses of C forming a partition of C (i.e., for each individual x of C, there is exactly one $i \in \{1, 2, \ldots, p\}$ such that x belongs to D_i), then $\texttt{mass}_C$ (resp., $\texttt{volume}_C$, $\texttt{number}_C$, etc.) is equal to the sum of the $\texttt{mass}_{D_i}$'s (resp., of the $\texttt{volume}_{D_i}$'s, of the $\texttt{number}_{D_i}$'s, etc.).

Finally, each variable v is assumed to satisfy the sign constraint $v \geqslant 0$.

The substitution (4) indicates that there should be neither egg nor armagnac in the adapted recipe. By contrast, there should be some bananas and kirsch but this piece of information can be entailed by the conservation equations. Therefore, the query is simply modeled by:

$$\texttt{Q} = (\texttt{mass}_{\texttt{Egg}} = 0) \wedge (\texttt{mass}_{\texttt{Armagnac}} = 0) \tag{8}$$

The adaptation problem is now formalized: the source case is formalized by (5); the query is formalized by (8) and the domain knowledge is given by the conversion and conservation equations, and the sign constraints. Since the source case and the query are to be understood wrt the domain knowledge, the formulas for them are, respectively, $\texttt{DK} \wedge \texttt{Source}$ and $\texttt{DK} \wedge \texttt{Q}$. The result of the adaptation will be denoted by AdaptedCase.

Description of the adaptation process. Let $\{v_1, v_2, \ldots, v_n\}$ be the set of the variables used in `Source`, `Q` and `DK`. In the representation space based on the formalism used above, a particular recipe is represented by a tuple $x = (x_1, x_2, \ldots, x_n) \in \Omega$, where $\Omega = \Omega_1 \times \Omega_2 \times \ldots \times \Omega_n$ such that $\Omega_i = \mathbb{Z}$ if $\tau(v_i) =$ `integer` and $\Omega_i = \mathbb{R}$ otherwise ($\mathbb{R}$: set of real numbers, $\mathbb{Z}$: set of integers). Given φ, a conjunction of linear constraints, let $\mathcal{M}(\varphi)$ be the set of $x \in \Omega$ such that x verifies all the constraints of φ. The function $\varphi \mapsto \mathcal{M}(\varphi)$ provides a model-theoretical semantics to the logic of the conjunction of linear constraints: φ_1 entails φ_2 if $\mathcal{M}(\varphi_1) \subseteq \mathcal{M}(\varphi_2)$.

The principle of revision-based adaptation consists in a minimal modification of $\texttt{DK} \wedge \texttt{Source}$ so that it becomes consistent with $\texttt{DK} \wedge \texttt{Q}$. Such a minimal modification can be computed thanks to a belief revision operator based on a distance function d on Ω, meaning that the modification from an $x \in \Omega$ to an $y \in \Omega$ is measured by $d(x, y)$. Let $S = \mathcal{M}(\texttt{DK} \wedge \texttt{Source})$ and $Q = \mathcal{M}(\texttt{DK} \wedge \texttt{Q})$. The minimal modification from the source case to the query is therefore measured by $d^* = d(S, Q) = \inf_{x \in S, y \in Q} d(x, y)$. Thus, `AdaptedCase` is such that

$$\mathcal{M}(\texttt{AdaptedCase}) = \{y \in Q \mid d(S, y) = d^*\}$$

where $d(S, y) = \inf_{x \in S} d(x, y)$.

Now, d is assumed to be a Manhattan distance function:

$$d(x, y) = \sum_{i=1}^{n} w_i |y_i - x_i|$$

where $w_i > 0$ is a weight associated to the variable v_i. Such a weight captures the effort of change for this variable. For example, if $v_i = \texttt{volume}_{\texttt{LemonJuice}}$ and $v_j = \texttt{volume}_{\texttt{Vodka}}$, then $w_i < w_j$ means that the adaptation process is less "reluctant" to change the volume of lemon juice than to change the volume of vodka.

Under this assumption, $\mathcal{M}(\texttt{AdaptedCase})$ is the solution of the following optimization problem in y:

$$x \in \mathcal{M}(\texttt{DK} \wedge \texttt{Source}) \qquad y \in \mathcal{M}(\texttt{DK} \wedge \texttt{Q}) \tag{9}$$

$$\text{minimize } d(x, y) \tag{10}$$

The conjunctions of constraints (9) are linear but the objective function (10) is not. Now, it can be shown that the set of solutions to this problem coincides with the set of solutions to the following optimization problem in y:

$$x \in \mathcal{M}(\texttt{DK} \wedge \texttt{Source}) \qquad y \in \mathcal{M}(\texttt{DK} \wedge \texttt{Q})$$

$$\bigwedge_{i=1}^{n} z_i \geq y_i - x_i \qquad \bigwedge_{i=1}^{n} z_i \geq x_i - y_i$$

$$\text{minimize } \sum_{i=1}^{n} w_i z_i$$

which is linear, and thus can be solved with classical operational research techniques. It is noteworthy that if every variable is continuous, then this optimization problem is polynomial, otherwise, it is a mixed integer linear optimization, known to be an NP-hard problem. In practice, the more variables are integers, the more it will require computing time; thus, if a variable range is big enough, it may be more appropriate to consider it as real. The heuristic we have chosen is as follows. If, for a type of food F, it appears in all the recipes of the case base as units, then $\tau(\texttt{number}_F) = \texttt{integer}$.

When this linear problem is solved, this gives a solution to the query, expressed with all the n variables. From a human-interface viewpoint, some of these variables should not be displayed. For example, if an ingredient is given by its volume in the source recipe, then it should not be given as a mass in the adapted case. Since DK relates masses to volumes, there is no loss of information.

With the example presented above, the result is as follows:

$$\begin{aligned}\texttt{AdaptedCase} \equiv\ & \texttt{DK}\\ & \wedge (\texttt{volume}_{\texttt{Kirsch}} = 9) \wedge (\texttt{volume}_{\texttt{Rum}} = 25) \wedge (\texttt{volume}_{\texttt{Milk}} = 50)\\ & \wedge (\texttt{number}_{\texttt{Banana}} = 2) \wedge (\texttt{mass}_{\texttt{GranulatedSugar}} = 96)\\ & \wedge (\texttt{volume}_{\texttt{FreshCream}} = 290)\end{aligned}$$

It can be noticed that AdaptedCase entails $\texttt{DK} \wedge \texttt{Q}$, which was expected. For this example, the following weights have been chosen assuming that more a variable correponds to a general concept more its associated weight has to be large:

$$w_{\texttt{volume}_{\texttt{Food}}} = 100 \qquad w_{\texttt{sugar}_{\texttt{Food}}} = 50 \qquad w_{\texttt{alcohol}_{\texttt{Food}}} = 50$$
$$w_{\texttt{volume}_{\texttt{Brandy}}} = 5 \qquad w_{\texttt{mass}_{\texttt{EggOrEquivalent}}} = 10$$
$$\text{and } w_v = 1 \text{ for any other variable } v$$

Translated back in an informal way, this gives:

AdaptedCase =	Recipe “Eggnog” (10 glasses) after adaptation
	9 cℓ of kirsch, 25 cℓ of rum, half a liter of milk, 2 bananas, 96 g of sugar, 290 cℓ of fresh cream

This result illustrates the quantity compensations done by the adaptation: the quantity of sugar has been lowered because bananas are sweeter than eggs and the volume of kirsch is higher than the volume of armagnac in the source recipe, because the degree of alcohol is lower for armagnac than for kirsch.

4 Sandwich challenge

The *sandwich challenge* is addressed with the 2014 Taaable system [9], which is efficient for the ingredient susbtitution step. The preparation procedure of the adapted recipe uses, in the same order, the steps used in the source recipe, because the ontology-based substitution procedure of Taaable favors the substitution of ingredients of the same type (e.g., a sauce by a sauce). So, the order of the ingredients in the adapted recipe will be the same as in the source recipe.

To adapt the textual preparation of the recipe, the text occurrences of the replaced ingredients are substituted with the replacing ingredients. A set of rules allows to identify plurals of the removed ingredient in the text, and replace them with the plural form of the replacing ingredients. For example, when replacing *mayo* with *mustard*, "Apply *mayo* on one slice, tomato sauce on the other." is adapted to "Apply *mustard* on one slice, tomato sauce on the other."

5 Conclusion

This paper has presented the two systems developed by the Taaable team for its participation to the 2015 CCC. The two systems are based on the previous version of Taaable, extended with two new approaches: a FCA approach to guide ingredient substitution, and an adaptation of the ingredient quantities based on a mixed linear optimization. The work presented here still needs a thorough evaluation: ongoing work addresses this issue, following the methodology introduced in [2].

References

1. A. Cordier, V. Dufour-Lussier, J. Lieber, E. Nauer, F. Badra, J. Cojan, E. Gaillard, L. Infante-Blanco, P. Molli, A. Napoli, and H. Skaf-Molli. Taaable: a Case-Based System for personalized Cooking. In S. Montani and L. C. Jain, editors, *Successful Case-based Reasoning Applications-2*, volume 494 of *Studies in Computational Intelligence*, pages 121–162. Springer, 2014.
2. E. Gaillard, J. Lieber, E. Nauer, and A. Cordier. How Case-Based Reasoning on e-Community Knowledge Can Be Improved Thanks to Knowledge Reliability. In *Case-Based Reasoning Research and Development*, volume 8765, pages 155 – 169, Cork, Ireland, Ireland, September 2014. L. Lamontagne and E. Plaza.
3. E. Gaillard, L. Infante-Blanco, J. Lieber, and E. Nauer. Tuuurbine: A Generic CBR Engine over RDFS. In *Case-Based Reasoning Research and Development*, volume 8765, pages 140 – 154, Cork, Ireland, September 2014.
4. E. Gaillard, J. Lieber, and E. Nauer. Adaptation knowledge discovery for cooking using closed itemset extraction. In *The Eighth International Conference on Concept Lattices and their Applications - CLA 2011*, pages 87–99, 2011.
5. B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, Berlin, 1999.
6. C. Carpineto and G. Romano. Effective Reformulation of Boolean Queries with Concept Lattices. In T. Andreasen, H. Christiansen, and H. Legind Larsen, editors, *Flexible Query Answering Systems, Third International Conference (FQAS'98)*, volume 1495 of *LNCS*, pages 83–94. Springer, 1998.
7. C. Carpineto and G. Romano. Order-Theoretical Ranking. *Journal of the American Society for Information Science*, 51(7):587–601, 2000.
8. J. Cojan and J. Lieber. Applying Belief Revision to Case-Based Reasoning. In H. Prade and G. Richard, editors, *Computational Approaches to Analogical Reasoning: Current Trends*, volume 548 of *Studies in Computational Intelligence*, pages 133 – 161. Springer, 2014.
9. E. Gaillard, J. Lieber, and E. Nauer. Case-Based Cooking with Generic Computer Utensils: Taaable Next Generation. In *Proceedings of the ICCBR 2014 Workshops*, number pp 89-100, page 254, Cork, Ireland, 2014. D. B. Leake and J. Lieber.

CookingCAKE: A Framework for the adaptation of cooking recipes represented as workflows

Gilbert Müller and Ralph Bergmann

Business Information Systems II
University of Trier
54286 Trier, Germany
[muellerg][bergmann]@uni-trier.de,
www.wi2.uni-trier.de

Abstract. This paper presents CookingCAKE, a framework for the adaptation of cooking recipes represented as workflows. CookingCAKE integrates and combines several workflow adaptation approaches applied in process-oriented case based reasoning (POCBR) in a single adaptation framework, thus providing a capable tool for the adaptation of cooking recipes. The available case base of cooking workflows is analyzed to generate adaptation knowledge which is used to adapt a recipe regarding restrictions and resources, which the user may define for the preparation of a dish.

Keywords: recipe adaptation, workflow adaptation, workflows, process-oriented case based reasoning

1 Introduction

Even after more than 30 years of research in CBR, adaptation is still a major challenge. This also applies to the adaptation of cooking recipes. Direct processing of textual recipes is however almost not feasible. Thus, they are usually transformed to structured cases, e.g., workflows [18]. In Process-Oriented Case-Based Reasoning (POCBR) [11], workflow adaptation is also an important research topic.

Existing methods for adaptation in CBR can be roughly classified into transformational, compositional, and generative adaptation [21,9]. While transformational adaptation relies on adaptations executed in a kind of a rule-based manner, generative adaptation demands general domain knowledge appropriate for an automated from scratch problem solver. An approach for transformational adaptation of workflows was presented by Minor et al. [10]. Compositional adaptation usually means that several cases are used during adaptation, incorporating transformational or generative adaptation methods involving adaptation knowledge. Dufour-Lussier et al. [4], for example, presented such a compositional adaptation approach. However, the different adaptation approaches come along with respective advantages and disadvantages. Thus, we expect that the integration and combination of several adaptation approaches can significantly improve the overall adaptation capability of a CBR system by overcoming some of the disadvantages of each individual approach.

In this paper, we present the next evolutionary step of your CookingCAKE system, which is the integration of three adaptation approaches developed within our previous research. In particular, we present a novel integration of adaptation by generalization and specialization, compositional adaptation, and transformational adaptation in POCBR. To achieve this, CookingCAKE analyzes the case base of cooking workflows generating an extensive adaptation knowledge base using several adaptation approaches. This knowledge is then used to adapt workflows according to the requirements and resources given in a current adaptation scenario. While this paper presents novel ideas and positions on adaptation, the *open challenge* is addressed. In addition, we present our examples as well as a comprehensive use case from the *sandwich challenge*, thus this challenge is addressed as well.

The next section introduces cooking workflows followed by a summary section sketching the used adaptation approaches from our previous research (see Sect. 3). Section 4 describes the novel integration of the approaches, including the generation of adaptation knowledge as well as the integrated adaptation itself. Next, Sect. 5 provides details on how the Computer Cooking Contest 2015 sandwich challenge is addressed. Finally, the paper wraps up by discussing potential future work.

2 Cooking Workflows

In our approach a cooking recipe is represented as a workflow describing the process to prepare a particular dish [18] (see Fig. 1). Cooking workflows consist of a set of *preparation steps* (also called *tasks*) and a set of *ingredients* (also called *data items*) shared between its tasks. Further, control-flow blocks may be used that represent either sequences, parallel (AND), alternative (XOR), or repeated execution (LOOPs) of preparation steps. These control-flow blocks may be nested but not interleaved, thus we consider block-oriented workflows only. This ensures the syntactic correctness of the workflow following the correctness-by-construction principle [17,3], e.g., that the workflow has one start node and one end node. Such workflows are referred to as consistent workflows. Tasks and control-flow blocks are linked by *control-flow edges* defining

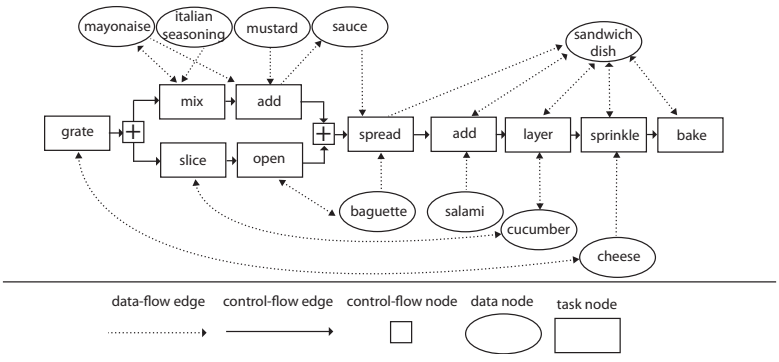


Fig. 1. Example of a block-oriented cooking workflow

the execution order. This forms the *control-flow*. Tasks, data items, and relationships (represented by *data-flow edges*) between the two of them form the *data flow*. An example block-oriented cooking workflow for a sandwich recipe is illustrated in Fig. 1.

2.1 Semantic Workflows and Semantic Workflow Similarity

To support retrieval and adaptation of workflows, the individual workflow elements are annotated with ontological information, thus leading to a *semantic workflow* [2]. CookingCAKE uses a taxonomy of ingredients to define the semantics of data items and a taxonomy of preparation steps to define the semantics of tasks. These taxonomies are employed for the similarity assessment between tasks and data items. An example ingredient taxonomy is given in Fig. 2. A taxonomy is ordered by terms that are either a generalization or a specialization of a specific other term within the taxonomy, i.e., an inner node represents a generalized term that stands for the set of most specific terms below it. For example, the generalized term $vegeterian$ stands for the set $\{potatoes, rice, noodles\}$. Further on in the paper we use inner nodes in generalized workflows to represent that an arbitrary ingredient from the set of its specializations can be chosen.

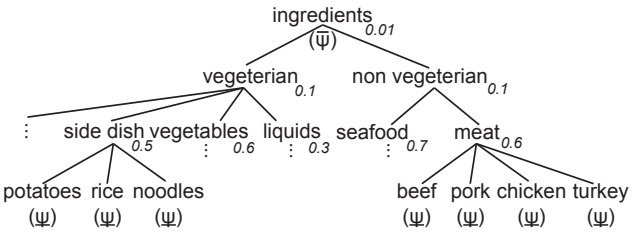


Fig. 2. Example of an ingredient taxonomy

In our previous work, we developed a semantic similarity measure for workflows that enables the similarity assessment of a case workflow W_c w.r.t. a query workflow W_q [2], i.e. $sim(W_q, W_c)$. Each query workflow element $x_q \in W_q$ is mapped by the function $m : W_q \rightarrow W_c$ to an element of the case workflow $x_c \in W_c$, i.e., $x_c = m(x_q)$. The mapping is used to estimate the similarity between the two workflow elements utilizing the taxonomy, i.e., $sim(x_q, x_c)$. The similarity of preparation steps or ingredients reflects the closeness in the taxonomy and further regards the level of the taxonomic elements. In general, the similarity is defined by the attached similarity value of the least common anchestor, e.g., $sim(beef, pork) = 0.6$. If a more general query element such as "meat" is compared with a specific element below it, such as "pork", the similarity value is 1. This ensures that if the query asks for a recipe containing meat, any recipe workflow from the case base containing any kind of meat is considered highly similar. All the similarity values of the mappings are then aggregated to estimate an overall workflow similarity.

2.2 Querying Semantic Workflows

In order to guide the retrieval and adaptation of workflows a query is defined by the user. CookingCAKE uses POQL (Query Language for Process-Oriented Case-Based Reasoning) [16] to capture desired and undesired ingredients or preparation steps of a cooking workflow as a query q. The definition of preparation steps is useful as certain tools might not be available or their usage is desired (e.g. oven). Let $q_d = \{x_1, \ldots, x_n\}$ be a set of desired ingredients or preparation steps and $q_u = \{y_1, \ldots, y_n\}$ be a set of undesired ingredients or preparation steps. A query q is then defined as $(x_1 \wedge \ldots \wedge x_2) \wedge \neg y_1 \wedge \ldots \wedge \neg y_n$. POQL also enables to capture generalized terms, i.e., if a vegeterian dish is desired, this can be defined by $\neg meat$. The query q is then used to guide retrieval, i.e., to search for a workflow which at best does not contain any undesired element and contains all desired elements. Based on the query q the unmatched elements can be identified, enabling estimating the elements to be deleted or added to the retrieved workflow during the subsequent adaptation stage. The similarity between the query and a workflow W is defined as the similarity between the desired ingredients and the workflow W and the number of undesired ingredients not contained in W according to the semantic similarity measure [2] in relation to the size of the query:

$$sim(q, W) = \frac{\sum_{x \in q_d} sim(x, m(x)) + |\{y \in q_u | sim(y, m(y)) \neq 1\}|}{|q_d| + |q_u|} \quad (1)$$

Hence, please note that similar desired ingredients or preparation steps increase the similarity while similar undesired ingredients or preparation steps do not reduce the similarity between the POQL query and the workflow.

In general, POQL is even more expressive and can, for example, capture time restrictions on preparation steps or that a certain ingredient should or should not be processed in a particular manner (e.g. do or do not bake vegetables). However, for the sake of simplicity we assume a set of desired and undesired ingredients or preparation steps only in the following sections.

3 Adaptation Approaches

This section summarizes the used adaptation approaches within the CookingCAKE framework.

3.1 Adaptation by Generalization and Specialization of Workflows

A generalized workflow [14] is a workflow containing generalized terms from a taxonomy (see Sec. 2.1), each of them representing multiple specialized ingredients or preparation steps. Thus, the generalized workflow represents a set of specialized workflows. Figure 3 illustrates an example for a generalization of the example workflow given in Fig. 1. Here, any preparation step that chops the cheese and any sort of meat could potentially be used. Such generalized workflows can be learned by comparing similar workflows from the case base. A workflow is generalized by generalizing terms if similar workflows from the case base contain several specializations of this generalized

term. It is assumed that if similar workflows contain the terms $\{beef, chicken, pork\}$, for example, these workflows can be generalized to contain any kind of $meat$. Likewise $makesmall$ represents all possible cooking steps reducing ingredients to small pieces.

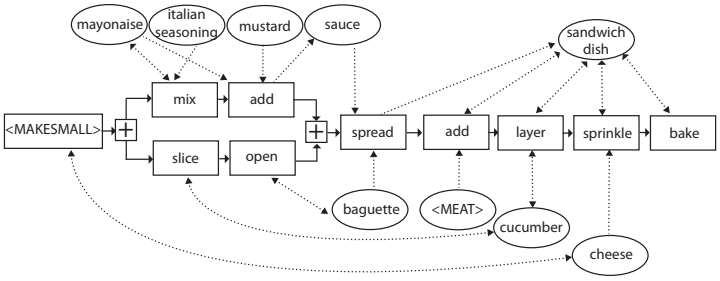


Fig. 3. Example of a generalized workflow

Adaptation is supported by specializing a workflow according to the POQL query q. Lets assume the generalized workflow contains the term $meat$ and the query defines that $beef$ is desired, the generalized element can be specialized according to $beef$. Thus, specialization enables adapting a workflow according to the POQL query.

3.2 Compositional Adaptation by Workflow Streams

The idea of compositional adaptation by workflow streams [13] is that each workflow can be decomposed into meaningful sub-components or snippets [7]. A sandwich workflow, for example, prepares the sauce and the toppings in order to produce the entire sandwich dish. These sub-components represented as partial workflows are referred to as *workflow streams*. Workflow streams can be identified by collecting all data-flow connected tasks[1] until a new data item such as sandwich sauce is created. An example for a workflow stream for the example workflow (see Fig. 1) is given in Figure 4 describing how to place toppings on the sandwich. To compute the adaptation knowledge, all workflow streams that can be found in the workflows within the case base are extracted.

The basic idea for compositional adaptation is, to adapt a workflow by using the workflow streams of other workflows that produce the same data item in a different manner, e.g., with other tasks or data. In the sandwich domain, for example, toppings, sauces, or preparation steps can be replaced. However, only workflow streams are substitutable if they produce the same data and consume identical data nodes. This ensures that replacing an arbitrary stream does not violate the semantic correctness of the workflow.

[1] If a task consumes a data item produced by another one, both tasks are dataflow-connected.

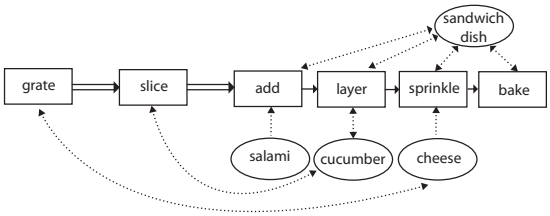


Fig. 4. Example of a workflow stream

3.3 Transformational Adaptation by Workflow Adaptation Operators

The workflow adaptation operators [15] are specified by two workflow sub-graphs called streamlets, one representing a workflow fraction to be deleted and one representing a workflow fraction to be added. Such operators can be learned from the case base by comparing two workflows and employ the difference between the two workflows in order to generate workflow adaptation operators. The example adaptation operator in Fig. 5 describes that mayonnaise can be replaced by tomatoes. This also enforces that tasks have to be changed as well, because the combine task also has to be exchanged for a chop task.

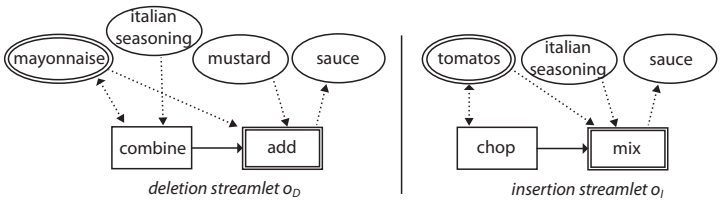


Fig. 5. Example of a workflow adaptation operator

The basic idea for operational adaptation is that chains of adaptation operators are applied $W \xrightarrow{o_1} W_1 \xrightarrow{o_2} \ldots \xrightarrow{o_n} W_n$ to the retrieved workflow W, thereby transforming the workflow W to an adapted workflow W_n. This process can be considered a search process towards an optimal solution w.r.t. the query. Hence, streamlets are removed, inserted, or replaced to transform the workflow according to the query.

4 CookingCAKE Framework

We now present the CookingCAKE framework which automatically generates adaptation knowledge using various adaptation approaches applied in POCBR (see Sect. 4.1). Based on this knowledge workflow adaptation is supported regarding a POQL query defining the requirements and resources on the workflow adaptation (see Sect. 4.2).

4.1 Generation of adaptation knowledge

As the acquisition of adaptation knowledge is an instance of the traditional knowledge acquisition bottleneck [5], CookingCAKE automatically generates adaptation knowledge based on the workflows contained in the case base (see Fig. 6). First, the case base and thus each workflow is generalized applying the method described in section 3.1. From this generalized case base further adaptation knowledge, i.e., workflow streams and adaptation rules (see Sect. 3), is automatically generated. As the adaptation knowledge is acquired based on the generalized case base, the adaptation knowledge itself is also generalized. This increases the adaptability for the entire adaptation procedure.

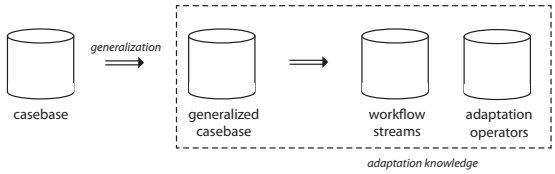


Fig. 6. Generation of adaptation knowledge

The generated adaptation knowledge can then be used to adapt a workflow whenever a query occurs. Further details on this procedure are explained in the next section.

4.2 Workflow adaptation

Whenever a POQL query occurs CookingCAKE searches for the workflow that best matches the given query within the generalized case base (see Fig. 7). However, it may happen that not all resources or requirements defined in the query are fulfilled by this workflow. Thus, workflow adaptation is required. For this purpose the workflow adaptation approaches presented in Sect. 3 are subsequently applied, still regarding the defined query. After this procedure, the adapted workflow still has to be specialized according to the query if it contains generalized elements. Therefore, CookingCAKE uses the specialization method presented in Sect. 3.1.

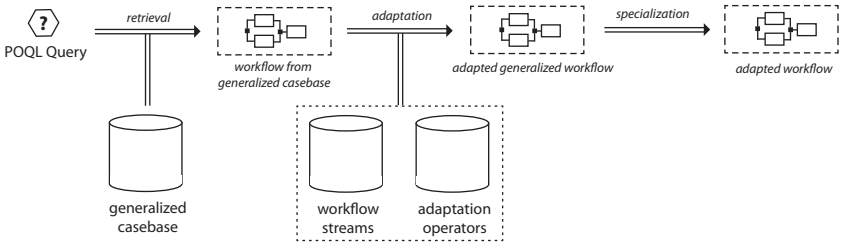


Fig. 7. Workflow adaptation

In order to ensure scalability of the presented approach for large case bases or large sets of adaptation knowledge, CookingCAKE supports a cluster-based retrieval [12] for workflows as well as for adaptation knowledge.

5 CCC Sandwich Challenge

In order to address the sandwich challenge a case base of 61 sandwich recipe workflows was created. The workflows were manually modelled based on sandwich recipes found on WikiTaaable[2] and further Internet sources. To enable similarity computations between the workflows, a modified version of the ingredient and cooking step ontology provided by WikiTaaable was employed. More precisely, multiple inheritance was resolved as CookingCAKE so far is only able to handle taxonomies (single inheritance). Further, the generalized terms of the taxonomies have been manually annotated with similarity values (see Sect. 2.1).

A running demo of CookingCAKE for the sandwich challenge is available under `http://cookingCAKE.wi2.uni-trier.de`[3] (see Fig. 8). The query of CookingCAKE contains desired and undesired ingredients as well as desired and undesired preparation

[2] http://wikitaaable.loria.fr

[3] Please note that CookingCAKE is still under improvement until the CCC'15

Fig. 8. Cooking Cake interface

steps. An example query (http://cookingCAKE.wi2.uni-trier.de?d=cherry%20tomato|salmon&u=cheese), generates a salmon and cherry tomato recipe without using any kind of cheese. Please note, that CookingCAKE does not necessarily fulfill the given query, it rather tries to fulfill the query as much as possible but does not execute any adaptations if no adaptation knowledge is present in order to remain the quality of the sandwich recipe. Consequently, if e.g. edam cheese is desired among other ingredients possibly a recipe with gouda cheese is returned if that is more suitable concerning the other desired desired ingredients. Further, undesired ingredients might be contained or a desired ingredient might not be contained, if that seems to be inappropriate according to the remaining ingredients given in the query. In general, cooking steps are adapted, if particular changed ingredients may require a different preparation of the particular dish.

After the definition of a query, CookingCAKE searches for the workflow in the case base that already best matches the given query based on the similarity value (see Sect. 2.2). If the query can not be fulfilled, adaptation is required. In this case, the entire adaptation procedure presented in Sect. 4.2 is used to adapt the sandwich recipe according to a query.

As a result, CookingCAKE can also print a detailed XML-File describing the used original case based recipe as well as the adapted recipe according to the query. Further, information is provided on which ingredients are removed from and added to the original workflow during adaptation.

Additionally, CookingCAKE also provides a textual view of the solution (see Fig. 9). For this purpose, the workflows are translated into a textual representation. Hence, the block-oriented workflow structure is reduced to a single sequence. Based on this, the required ingredients and the sequence of preparation steps (including the information on which ingredients are required in every preparation step) are generated. Further, the workflow itself is also illustrated in the process view.

CookingCAKE also features a name generator for the generated recipes. It accesses the taxonomy of ingredients and combines several terms of sub-taxonomies contained as ingredients in the workflow to assign a name to a recipe.

Based on the 61 recipes stored in CookingCAKE, generalization and specialization enable to generate more than $9 \cdot 10^{21}$ recipes. Further adaptations are supported by 197 workflow streams found and 7870 operators (1306 replace, 3903 insert, 2661 delete) generated. As the streams and operators are also generalized (see Sect. 4.2) adaptability is further increased. Hence, CookingCAKE provides a capable tool for the adaptation of sandwich recipes.

6 Conclusions and Future Work

We presented CookingCAKE, a framework for the adaptation of cooking recipes represented as workflows integrating and combining various adaptation approaches applied in Process-Oriented Case-Based Reasoning (POCBR). The available case base of cooking workflows is analyzed to generate adaptation knowledge which is used to adapt a recipe regarding a given query for the preparation of a dish.

In future work, we will investigate and integrate additional adaptation approaches for workflows such as the abstraction of workflows containing abstract tasks (e.g., pre-

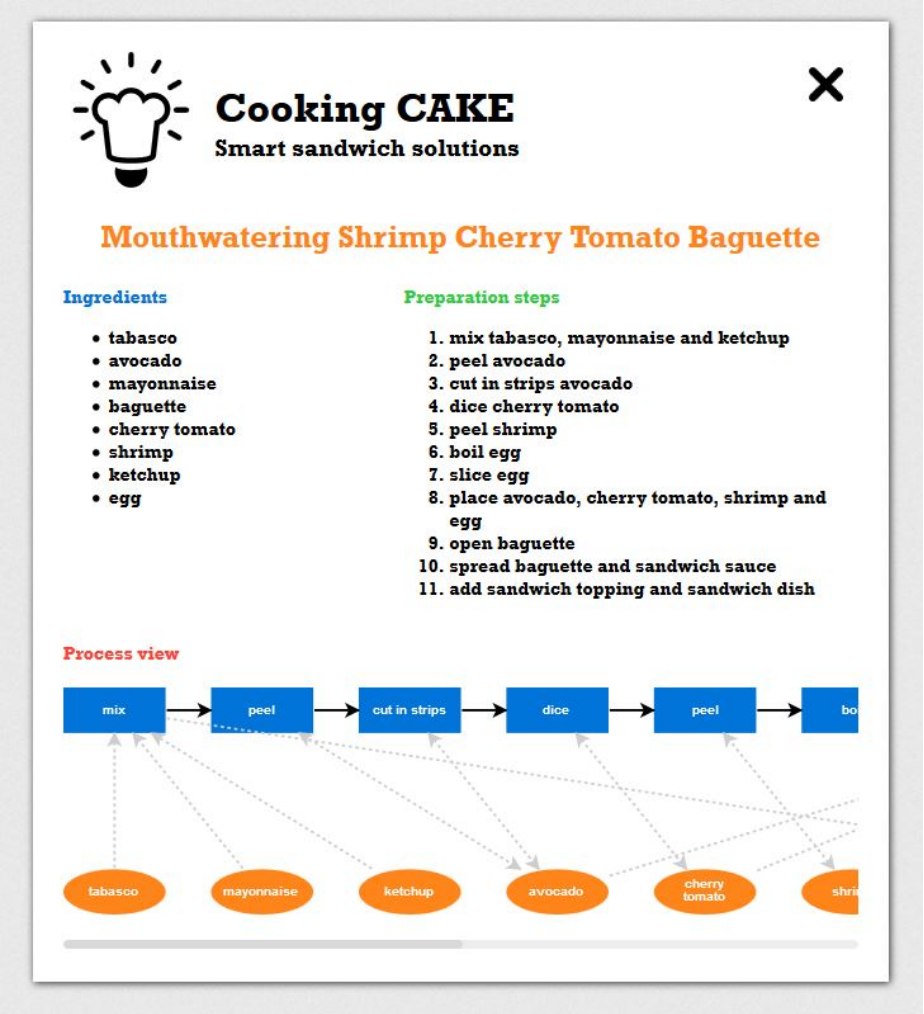


Fig. 9. Example recipe generated by CookingCAKE

pare sauce, place toppings on sandwich). Further, we will integrate the case-based adaptation approach of Minor et al. [10] in our framework. Moreover, CookingCAKE will be extended to be able to handle more knowledge-intensive ontologies, e.g., ontologies with multiple inheritance. Future work will also comprise the retrieval of adaptable cases [19], i.e., we will investigate the adaptability of the workflows within the case base as the workflow that best matches the given query is not necessarily the workflow that can be at best adapted to the resources and requirements given. Consequently, a better workflow as starting point for the adaptation can be chosen. Moreover, the retainment of adaptation knowledge[6] will be addressed by gathering user feedback on the adapted cooking recipes. This is important, as the quality of automatically learned

adaptation knowledge can not always be ensured. Thus, the quality of workflow adaptation is improved and the growth of adaptation knowledge can be controlled. Finally, CookingCAKE will be extended by interactive adaptation [1,8,20]. This supports the search of a suitable query by involving user interaction during adaptation which assist the user to create more individual cooking recipes.

Acknowledgements. This work was funded by the German Research Foundation (DFG), project number BE 1373/3-1.

References

1. Aha, D.W., Muñoz-Avila, H.: Introduction: Interactive case-based reasoning. Applied Intelligence 14(1), 7–8 (2001)
2. Bergmann, R., Gil, Y.: Similarity assessment and efficient retrieval of semantic workflows. Inf. Syst. 40, 115–127 (Mar 2014)
3. Dadam, P., Reichert, M., Rinderle-Ma, S., Göser, K., Kreher, U., Jurisch, M.: Von ADEPT zur AristaFlow BPM Suite-Eine Vision wird Realität:" Correctness by Construction" und flexible, robuste Ausführung von Unternehmensprozessen (2009)
4. Dufour-Lussier, V., Lieber, J., Nauer, E., Toussaint, Y.: Text adaptation using formal concept analysis. In: Bichindaritz, I., Montani, S. (eds.) Case-Based Reasoning. Research and Development, LNCS, vol. 6176, pp. 96–110. Springer (2010)
5. Hanney, K., Keane, M.: The adaptation knowledge bottleneck: How to ease it by learning from cases. In: David Leake, E.P. (ed.) Case-Based Reasoning Research and Development 1997, pp. 179–192. LNAI 1266, Springer (1997)
6. Jalali, V., Leake, D.: On retention of adaptation rules. In: Lamontagne, L., Plaza, E. (eds.) Case-Based Reasoning Research and Development, Lecture Notes in Computer Science, vol. 8765, pp. 200–214. Springer International Publishing (2014)
7. Kolodner, J.L. (ed.): Proceedings Case-Based Reasoning Workshop. Morgan Kaufmann Publishers, San Mateo, California (1988)
8. Leake, D.B., Wilson, D.C.: Combining CBR with interactive knowledge acquisition, manipulation and reuse. In: Case-Based Reasoning Research and Development, pp. 203–217. Springer (1999)
9. Lopez Mantaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M.L., Cox, M.T., Forbus, K., Keane, M., Aamodt, A., Watson, I.: Retrieval, reuse, revision and retention in case-based reasoning. The Knowledge Engineering Review 20(03), 215–240 (2005)
10. Minor, M., Bergmann, R., Görg, S., Walter, K.: Towards case-based adaptation of workflows. In: Case-Based Reasoning. Research and Development, pp. 421–435. Springer (2010)
11. Minor, M., Montani, S., Recio-Garca, J.A.: Process-oriented case-based reasoning. Information Systems 40(0), 103 – 105 (2014)
12. Müller, G., Bergmann, R.: A cluster-based approach to improve similarity-based retrieval for Process-Oriented Case-Based Reasoning. In: 20th European Conference on Artificial Intelligence (ECAI 2014), IOS Press (2014)
13. Müller, G., Bergmann, R.: Workflow Streams: A Means for Compositional Adaptation in Process-Oriented Case-Based Reasoning. In: Proceedings of ICCBR 2014. Cork, Ireland (2014)
14. Müller, G., Bergmann, R.: Generalization of Workflows in Process-Oriented Case-Based Reasoning. In: 28th International FLAIRS Conference. AAAI, Hollywood (Florida), USA (2015)

15. Müller, G., Bergmann, R.: Learning and Applying Adaptation Operators in Process-Oriented Case-Based Reasoning. In: Proceedings of ICCBR 2015. Frankfurt, Germany (2015)
16. Müller, G., Bergmann, R.: POQL: A New Query Language for Process-Oriented Case-Based Reasoning. In: Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB. COER Workshop Proceedings, to appear (2015)
17. Reichert, M.: Dynamische Ablaufänderungen in Workflow-Management-Systemen (2000)
18. Schumacher, P., Minor, M., Walter, K., Bergmann, R.: Extraction of procedural knowledge from the web. In: Workshop Proceedings: WWW'12. Lyon, France (2012)
19. Smyth, B., Keane, M.: Retrieving adaptable cases. In: Wess, S., Althoff, K.D., Richter, M. (eds.) Topics in Case-Based Reasoning, Lecture Notes in Computer Science, vol. 837, pp. 209–220. Springer Berlin Heidelberg (1994)
20. Weber, B., Wild, W., Breu, R.: CBRFlow: Enabling adaptive workflow management through conversational case-based reasoning. In: Advances in Case-Based Reasoning, pp. 434–448. Springer (2004)
21. Wilke, W., Bergmann, R.: Techniques and knowledge used for adaptation during case-based problem solving. In: Tasks and Methods in Applied Artificial Intelligence, pp. 497–506. Springer (1998)

«CooCo, what can I cook today? Surprise me.»

Karen Insa Wolf, Stefan Goetze, and Frank Wallhoff

Fraunhofer Institute for Digital Media Technology IDMT
Marie-Curie-Strasse 2, 26129 Oldenburg, Germany
{insa.wolf,stefan.goetze,frank.wallhoff}@idmt.fraunhofer.de
http://www.idmt.fraunhofer.de

Abstract. In this paper a heuristic computer-based approach is described to vary cooking recipes by replacing ingredients. Conceptually, the approach is integrated in a speech dialogue system. The approach is based on a scoring system. The score value is used to rate different ingredients as candidates to substitute a specific ingredient of a recipe. This substitution score depends on different factors: 1) rating of the similarity between the ingredient which has to be replaced and the substitution candidate 2) rating how well the substitution candidate fits the recipe 3) gustatory preferences of the user. The substitution candidate with the highest score is proposed to the user.

Keywords: speech dialogue system, cooking coach, recipe variation

1 Motivation

The task in the open challenge of the Computer Cooking Contest [1] is a computer-based adaptation of cooking recipes. The present contribution proposes an approach to substitute ingredients of recipes. The approach is integrated in a speech dialogue system, called CooCo (Cooking Coach), introduced in [13]. CooCo is currently being further developed. A speech dialogue system is a suitable framework for this task:

- Speech input and output is a natural and convenient way to interact with technical devices or systems.
- A speech dialogue system is particularly suitable in scenarios in which the user cannot use his or her hands for interaction. Keyboard, mouse or touchscreen are not convenient user interfaces while cooking.
- Assuming a flexible dialogue management, spontaneous utterance of the user (like e.g. «Oops, I do not have ...») can be processed.
- The user can be involved in a unobtrusive manner to improve the recipe variation result and tailor the recipe to her/his personal gusto.

2 Concept of CooCo

CooCo is designed to assist users in different scenarios: The user can ask for recipes while doing the dishes or can get reminders regarding timing and next

steps while cooking. Both tasks require a context-based dialogue system including modules for interpreting, planning and re-planning, as well as memorizing and learning. Different approaches to realize a speech dialogue manager exist, e.g. [9]. Lison distinguishes between hand-crafted and statistical approaches and proposes the toolkit OpenDial to combine both [8]. The dialogue manager of CooCo is based on OpenDial [10]. CooCo's assistance while cooking is conceptually based on a dynamic planning module to actively manage the cooking process. This goes beyond simply reading out the cooking steps aloud when the user asks for this [11]. CooCo formulates an action plan considering active and passive time of the user (e.g. cutting vs. simmering) and dependencies of the cooking steps [13]. The recipe advice mode includes generic models of gustatory preferences (e.g. hot or sweet depending on typical amount of ingredients like chili or sugar) which will be adapted based on the feedback of the user. A new feature, presented in this paper, is the variation of the cooking recipes.

3 Computer-based variation of cooking recipes

The computer-based variations of cooking recipes addresses topics of artificial intelligence and machine learning approaches. The task to derive the consequences of the substitution of an ingredient on the textual description of the preparation steps requires techniques of natural language understanding, e.g. [2]. Other approaches aim at replacing ingredients, e.g. by randomizing recipe items [3], by using cognitive super computing (based on IBM's computer system WATSON, [6]) or by just enlarging the database (by the help of a community) to find a matching recipe for every combination of ingredients [12].

The approach presented here addresses the replacement of ingredients. Thereby, I_{db} is the set of all ingredients (I) of a specific database. A subset $I_{rc} \subseteq I_{db}$ with ingredients, which belong to one recipe, is defined as $I_{rc} = \{i_{rc,1}, \ldots, i_{rc,m}\}$ with maximum number m of ingredients. The subset $I_{sb} = \{i_{sb,1}, \ldots, i_{sb,h}\}$ with $h \leq m$ and $I_{sb} \subseteq I_{rc}$ comprises all ingredients which will be substituted. The food items which are candidates (C) to substitute one element of I_{sb} belong to the set $C_{sb} = \{c_{sb,1}, ..., c_{sb,n}\}$ with maximum number n of known food items. The set of the remaining ingredients of the recipe without the elements of I_{sb} is defined as $I_{rm} = I_{rc} \setminus I_{sb}$. The approach is based on the computation of a substitution score s ranging from 0 to 120 indicating the fit of a specific substitution pair $i_{sb,j} \in I_{sb}$ and $c_{sb,k} \in C_{sb}$. The substitution score is based on statistical information derived from a recipe database and general food knowledge. The approach can also be regarded as one module of a case-based reasoning process of a recipe advisor, as it is described e.g. in [7], to include the substitution of ingredients.

4 Use cases

The central task in the following two use cases is to propose a tasty recipe based on the user's input by replacing ingredients. The intention of the user differs in

the scenarios. Both use cases can be extended by including the question of undesired ingredients. In order to enlarge the number of possible recipe candidates, the proposed recipe variation approach can be applied in this case additionally to substitute undesired ingredients. Users differ in their gustatory preferences, one likes more traditional recipes, while the other is more open to new tastes. To adjust these individual preferences, two user parameters are introduced referred to as experimental levels. The experimental level e_{cd} influences how common or uncommon a substitution candidate should be. The level e_{cb} regulates how common or uncommon the combination of a substitution candidate and all elements of I_{rm} is. For both levels three adjustment steps can be chosen by the user, ranging from 1 = very common to 3 = very uncommon.

4.1 Use case 1: ≪Suprise me.≫

Based on one chosen recipe the user asks for a variation of this recipe. A similar scenario would be that the user realizes that one ingredient is missing but s/he still wants to cook the chosen recipe accepting variations. In both cases, CooCo can choose freely possible substitution candidates. In the first case, the ingredient $i_{sb,j}$ is not defined by the user. In the second case, $i_{sb,j}$ is the missing ingredient.

4.2 Use case 2: ≪Work with what I have.≫

The user specifies some ingredients I_{us}, s/he wants to work with, but no recipe can be found in the database which uses all desired ingredients. The task for CooCo is now to propose one recipe which matches by replacing missing elements (I_{ms}) of I_{rc} with those of I_{us}. For this scenario, a plausibility check is necessary since not each combination of ingredients presents a suitable option for a recipe.

5 CooCo's Recipe Variation Approach

The central aim of the approach is to compute substitution scores s for different substitution candidates of C_{sb} in relation to one element $i_{sb,j}$ of I_{sb}. The candidate $c_{sb,k}$ with the highest score is finally proposed to the user. Considering the abbreviations $i_{sb,j} = i$ and $c_{sb,k} = c$ the substitution score $s(i, c)$ is derived as

$$s(i,c) = s_b(i,c) + s_{sp}(i,c) + s_n(i,c) + s_{cd}(c) + s_{cb}(c, i_{rm}|i_{rm} \in I_{rm}), \qquad (1)$$

with s_b as basic substitution score, s_{sp} as special substitution score, s_n as substitution score based on nutrition facts, and s_{cd} and s_{cb} as substitution scores derived from a statistical analysis of the ingredients and their combination frequency based on the recipe database. The derivation of each summand of Eq. 1 is explained in the following. The substitution of more than one ingredient can be done by repeating the algorithm, up to now without considering the results of subsequent substitution steps. As starting point a recipe database with 1.222 recipes is chosen [5]. Additionally, a semantic net is created representing food

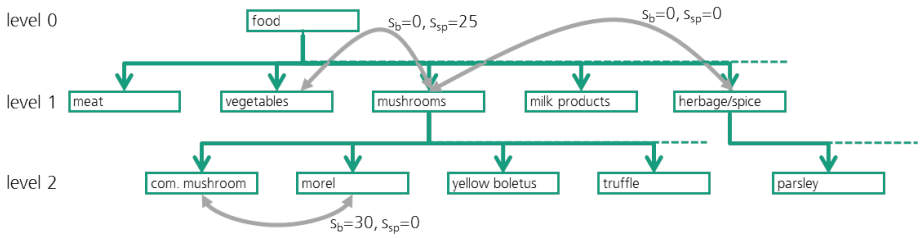


Fig. 1. Part of the semantic net with substitution scores s_b and s_{sp}.

items in a structured way, cf. Fig. 1. Each item is represented as class within a relationship network of currently 120 classes starting from the level 0 up to 3.

Besides the parent-children constellation different properties of each food class are stored. These properties are grouped in (a) those properties considering only the class itself and (b) those properties related to other classes. For group (a), the following properties are introduced:

nutrition facts n_g, **with** $g = \{c, f, p, e\}$**:** Nutrition facts are stored for different food classes of level 2 or higher. In the first version of CooCo, the variable n_c contains carbohydrates, n_f fat,n_p protein, and n_e energy per 100 g.

relative frequency f_{cd}**:** For each food class its relative frequency is derived based on the recipe database. The number of recipes in which the class occurs as ingredient is divided by the total number of recipes. This frequency value describes how common or uncommon a certain ingredient is.

substitution score s_{cd}**:** Based on the relative frequency f_{cd} the score s_{cd} is derived, considering the experimental level e_{cd}. The relative frequencies f_{cd} are classified in five categories. The first category $D_{cd,1}$ contains rarely used and the last category $D_{cd,5}$ frequently used ingredients, assuming $D_{cd} := [0 \ldots 0.005 \ldots 0.01 \ldots 0.03 \ldots \ 0.08 \ldots 1.0]$. The index of the category in combination with the experimental level e_{cd} defines the magnitude of the substitution score s_{cd}. This is implemented using a weighting matrix

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} \\ w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \\ w_{31} & w_{32} & w_{33} & w_{34} & w_{35} \end{bmatrix} = \begin{bmatrix} -2 & -1 & 0 & 1 & 2 \\ 0 & 0 & 0 & 0 & 0 \\ 2 & 1 & 0 & -1 & -2 \end{bmatrix} \tag{2}$$

and by taking one of its elements to derive

$$s_{cd} = 10 w_{e_{cd},p}, \tag{3}$$

with row number e_{cd} and column number p as index of $D_{cd,p}$.

The group (b) of properties considers the relation between two food classes to indicate how good they can substitute each other or how good they can be combined in one recipe.

basic substitution score s_b: It is assumed that food classes at a low semantic level (e.g. common mushrooms, morel or truffel in the class "mushrooms", level 2) are similar to each other. Therefore, the score s_b is introduced depending on the level of class in the semantic net, cf. Fig. 1.

special substitution score s_{sp}: A few explicitly defined scores s_{sp} are stored (e.g. vegetables as substitution candidate for mushrooms or the milk product "tofu" as good substitution candidate for children of the class "meat").

substitution score based on nutrition facts s_n: It is assumed that one food class of level 2 or higher is a good substitution candidate for another food class if they have similar nutrition facts. Therefore, the similarity factor f_n of two ingredients i_A and i_B is derived based on their nutrition facts n_g as

$$f_{n,g}(i_A, i_B) = \frac{|n_g(i_A) - n_g(i_B)|}{(n_g(i_A) + n_g(i_B))}, \tag{4}$$

with $g = \{c, f, p, e\}$. The mean value μ_{f_n} and the standard deviation σ_{f_n} derived from all $f_{n,g}$ is used as measure of the similarity of the nutrition facts - being aware of the roughness and simplicity of this approach. The substitution score based on nutrition facts is derived as

$$s_n = \min(2/\mu_{f_n}, 15) + \min(2/\sigma_{f_n}, 15). \tag{5}$$

relative combination frequency f_{cb}: The frequency value $f_{cb}(i_A, i_B)$ expresses how often an ingredient i_A is used in combination with a specific ingredient i_B of I_{db}. Therefore, the number of recipes $n_{A\&B}$, in which both ingredients i_A and i_B are included, is determined. This yields $f_{cb}(i_A, i_B) = n_{A\&B}/n_A$. As the denominator usually differs numerically for $f_{cb}(i_A, i_B)$ and $f_{cb}(i_B, i_A)$, the frequencies differ correspondingly. Following this approach, it has to be considered that uncommon ingredients can get f_{cb} values close to 1.0 as n_A is close to $n_{A\&B}$.

substitution score s_{cb}: The score s_{cb} is derived in a similar way as it is done for s_{cd}, but now considering the relative frequency f_{cb} and the experimental level e_{cb}. All frequencies $f_{cb}(c, i)$ of the substitution candidate $c \in C_{sb}$ and the ingredients $i \in I_{rem}$ have to be calculated first. The mean value of all these frequencies is then derived as $f_{cb}(c, I_{rm})$. This value is finally assigned to one of the categories $B_{cb,1}$ up to $B_{cb,5}$, heuristically defined as $B_{cb} := [0 \ldots 0.30 \ldots 0.35 \ldots 0.40 \ldots 0.50 \ldots 1.0]$. Using the weighting matrix W from Eq. 2 the score is derived as

$$s_{cb} = 10 w_{e_{cb},q}, \tag{6}$$

with row number e_{cb} and column number q as index of $B_{cb,q}$. The effect is, that if the user wants uncommon combinations, expressed as $e_{cb} = 3$, a set of ingredients with a low $f_{cb}(c, I_{rm})$ get a high score.

6 Implementation

The knowledge base containing the recipe database and the semantic net are implemented in Prolog. Standard request functions are implemented, so that recipes

including or excluding specific ingredients can be looked up. The approach described above is conceptually tested, further implementation and evaluation is on going work. The procedures to handle both use cases (cf. Section 4) are described in the following based on one test example. The starting point is a simple mushroom soup recipe:

250 g common mushrooms,	40 g butter,	40 g flour,
5 dl bouillon,	5 dl milk,	1 tb parsely, minced,
- - salt,	- - pepper	

6.1 Procedure for use case 1: ≪Surprise me.≫

In the following description only some examples of the possible substitution candidates are listed.

1. Choose the ingredient with the largest proportion relative to all ingredients of I_{rc} as i_{sb} [*common mushrooms*].
2. Compute $(s_b + s_{sp})$ for all $c_{sb} \in C_{sb}$ [class "mushrooms": *yellow boletus, morel, truffle*; extract of class "vegetables": *red pepper, tomato, cucumber*]
3. Compute s_n for all pairs of c_{sb} and i_{sb} [listing *parsley, cauliflower, morel, yellow boletus* as the one with a high score s_n].
4. Derive s_{cd} for all c_{sb}.
5. Derive s_{cb} for all c_{sb} with respect to the elements of I_{rem}.
6. Sum up s for each pair of c_{sb} and i_{sb} following Eq. 1.

Numerical results for the different experimental levels e_{cd} and e_{cb} are listed in Tab. 1. Parsley is left out as it is already part of the recipe. The results show, that a user with a low e_{cd} of 1 and a medium or high e_{cb} of 2 or 3 will be recommended a tomato soup. In case a very common combination of ingredients is wanted $(s(1,1))$, morel soup is proposed instead. Reason for this is that the recipes with common mushrooms and morel often share the basic combination of ingredients. A user who wants uncommon ingredients in a uncommon combination gets truffle as substitution candidate $(s(3,3))$. Elements of the class mushrooms are mostly preferred. A whole class like "mushrooms" could also be excluded, resulting in recommendations of cauliflower as substitution candidate as a less common ingredient than tomatoes. As the terms s_{cd} and s_{cb} are based on the statistical analysis of the recipe database, the result depends strongly on the size and the quality of the recipe database.

6.2 Procedure for use case 2: ≪Work with what I have.≫

In use case 2 the user desires a recipe with the ingredient set $I_{us} = \{$*butter, flour, parsley, bouillon, red pepper*$\}$. Firstly, for all elements of I_{us} the frequencies f_{cb} to each other are checked heuristically: If all $f_{cb} > 0$ and the mean $\mu_{f_{cb}} > 0.1$, then CooCo accepts the set I_{us}. Otherwise the dialogue with the user is reopened to ask for other set members. In the example, the set is accepted. The recipe that matches best I_{us} is mushroom soup, based on the simple rule to look for

Table 1. Numerical results of use case 1. The result $s(j,k)$ means s based on $e_{cd} = j$ $e_{cb} = k$. The respective candidate with the largest score s is marked in bold letters.

	y. boletus	morel	truffle	red pepper	tomato	cucumber	cauliflower
$s_b + s_{sp}$	30	30	30	25	25	25	25
s_n	29.2	29.2	29.0	9.4	5.6	12.8	33.0
f_{cd}	0.007	0.004	0.002	0.107	0.142	0.010	0.010
f_{cb}	0.397	0.543	0.286	0.347	0.346	0.236	0.367
$s(1,1)$	49	**59**	19	44	51	8	48
$s(1,2)$	49	39	39	54	**61**	28	48
$s(1,3)$	49	19	59	64	**71**	48	48
$s(2,1)$	59	**79**	39	24	31	18	58
$s(2,2)$	59	**59**	**59**	34	41	38	58
$s(2,3)$	59	39	**79**	44	51	58	58
$s(3,1)$	69	**99**	59	4	11	28	68
$s(3,2)$	69	**79**	**79**	14	21	48	68
$s(3,3)$	69	59	**99**	24	31	68	68

those recipes with the smallest number of missing ingredients $I_{ms} = \{i_{ms} | (i_{ms} \in I_{rc}) \wedge (i_{ms} \notin I_{us})\}$. However, red pepper is not part of the original recipe. The algorithm now attends to compute based on Eq. 1 as criteria whether red pepper is a suitable substitution candidate c_{sb} for one of the missing ingredients. Some of the missing ingredients {*pepper, salt, bouillon*} are marked as standard ingredients in the database. CooCo assumes as first guess that they are available also in case the user did not mentioned them explicitly. If this is confirmed by the user, the only missing ingredients left are $I_{ms} = \{$*common mushrooms, milk*$\}$. Considering the experimental levels, the score s is derived for all pairs of c_{sb} with one of the elements of I_{ms}. The computation result looking at the substitution pair *red pepper - common mushrooms* differs slightly compared to the result of use case 1 because *milk* is left out in the computation of s_{cb} as it is missing. As consequence, f_{cb} is classified here in $B_{cb,3}$ resulting in $s_{cb} = 0$, independently of e_{cb} as the weight factor in Eq. 6 is zero. Therefore, for all e_{cb} levels the score s is identical to $s(j,2)$ with $e_{cd} = j$, cf. Tab. 1. The highest score $s = 54$ is reached for $e_{cd} = 1$. Considering a threshold scheme of $[120 \ldots 80]$ (very good), $]80 \ldots 40]$ (acceptable), $]40 \ldots 0]$ (not recommended) for s, the substitution pair *red pepper - common mushrooms* is evaluated as "acceptable". In no case it is an option to replace *milk* with *red pepper*, the highest score is $s = 29$. This is reasonable, but a rule should be added in future versions to avoid the substitution of liquid and solid ingredients in any case. This is possible by adding an appropriate property in the semantic net. Milk remains here as missing candidate. Two different last options are possible: (1) Ask the user explicitly whether there is after all a potential substitution candidate. If yes, repeat the procedure. (2) Evaluate how well the missing ingredient could be omitted. Therefore, $s_b + s_{sp} + s_n$ is computed in relation to all ingredients of I_{rm} to get a hint if one of them could make up for the omission by increasing its quantity. In this specific example, the result of 17.5 for *milk* in relation to *butter* is not promising enough to propose this

as solution. As final step, the amount of liquid within the recipe ingredients is checked leading here to an increase of the amount of bouillon to recover the original amount of liquid. The final solution with appropriate comments based on the score s is presented to the user.

7 Conclusion and future work

A new feature of the currently developed application CooCo is presented. An approach to derive recipe variations by replacing ingredients is introduced. Two different use cases are addressed. The introduced examples provide reasonable results. This first proposed version of the approach has to be further improved and expanded in future work. An evaluation of the substitution results is planned based on feedback of users integrated in the speech dialogue system. The mechanism how to choose the best starting recipe in use case 2 can be ameliorated, including e.g. more information of the gustatory preferences of the user. The present approach prefers recipes with a small number of ingredients. In summary, the approach is a first step for computer-based tasty cooking recipe variations.

Acknowledgement. The authors would like to thank the anonymous reviewers for their helpful comments to improve the final version of the paper and their suggestions for future work in this research field.

References

1. CCC: Computer cooking contest (2015), `cc2015.loria.fr/index.php`, 3.3.2015
2. Dufour-Lussier, V., Le Ber, F., Lieber, J., Nauer, E.: Automatic case acquisition from texts for process-oriented case-based reasoning. Inf Systems 40, 153–167 (2014)
3. Easier Baking, `www.easierbaking.com`, 9.7.2015
4. Gamrad, D.: Modeling, simulation, and realization of cognitive technical systems. Ph.D. thesis, Universität Duisburg-Essen (2011)
5. Herz, J.: Kalorio (2015), `www.kalorio.de/`, 27.2.2015
6. IBM, Inst of Culinary Education: Cognitive Cooking with Chef Watson: Recipes for Innovation from IBM & the Institute of Culinary Education. Sourcebooks (2015)
7. Ihle, N., Newo, R., Hanft, A., Bach, K., Reichle, M.: CookIIS - A Case-Based Recipe Advisor. In: Proc 8th Int Conf on CBR 2009, Seattle, USA, pp. 269-278 (2009)
8. Lison, P.: Structured Probabilistic Modelling for Dialogue Management. Ph.D. thesis, Dep Informatics, University of Oslo (2014)
9. Morbini, F., Audhkhasi, K., Sagae, K., Artstein, R., Can, D., Georgiou, P., Narayanan, S., Leuski, A., Traum, D.: Which ASR should I choose for my dialogue system? In: Proc SIGDIAL 2013, Metz, France. pp. 394 – 403 (2013)
10. Open Dial, `www.opendial-toolkit.net`, 9.7.2015
11. Schäfer, U., Arnold, F., Ostermann, S., Reifers, S.: Ingredients and recipe for a robust mobile speech-enabled cooking assistant for german. In: KI 2013: Advances in Artificial Intelligence, pp. 212–223. No. 8077 in LNCS, Springer (2013)
12. SousChef, `www.acaciatreesoftware.com/`, 9.7.2015
13. Wolf, K.I., Goetze, S., Wellmann, J., Winneke, A., Wallhoff, F.: Concept of a nutrition consultant application with context based speech recognition. In: Proc Workshop Kognitive Systeme 2015, Bielefeld (2015)

Enriching Cooking Workflows with Multimedia Data from a High Security Cloud Storage

Patrick Bedué | bedue@stud.uni-frankfurt.de
Wenxia Han | s0611400@stud.uni-frankfurt.de
Mathias Hauschild | s8722400@stud.uni-frankfurt.de
Maximilian Pötz | s8084343@stud.uni-frankfurt.de
Mirjam Minor | minor@cs.uni-frankfurt.de

Abstract: With increasing growth of cloud services and the ability to choose from different cloud providers, we propose a new way to connect cooking workflows with a high security cloud storage. We use Activiti for workflow design and JavaScript Object Notation (JSON) for structured data interchange with a sealed cloud storage. This approach supports cooking workflows with instructions from multimedia data (e.g. videos, pictures) for special interest groups of the cooking domain like private communities or even chronically ill patients. The paper makes a contribution to current trends in information-systems related research such as scalability and experience reuse. Further, it connects Cloud Computing with Business Process Management.

1 Introduction

With growing globalization, information technology has become a key resource for business success or failure. IT is often used to manage business processes in companies and has become increasingly important, leading to a rise in new ways to organize business processes (Aalst, Benatallah et al. 2003). Cloud Computing changes the way we can develop and organize our resources and also enables a flexible and individual allocation of resources (Ciovică, Cristescu et al. 2014, Schulte, Janiesch et al. 2015).

Business processes are modelled as a collection of activities, dependencies between activities and are technically supported by workflows (Aalst, Benatallah et al. 2003). Regarding actual research topics, realizing business processes in a flexible and cost-efficient way is on a rise (Schulte, Janiesch et al. 2015). There are some studies focusing on workflow execution in the cloud investigating infrastructural challenges of elastic Business Process Management or security issues (Wang, Korambath et al. 2014, Schulte, Janiesch et al. 2015).

To develop our research proposition, we concentrate our work on the implementation of workflows in the cooking domain with a special focus on applying secure cloud technology for multimedia data integration and data objects representing ingredients.

To improve the user experience, we investigate the feasibility of using multimedia data from a high security cloud storage and of integrating this data into workflows. The user advantage is that she no longer needs to store big multimedia files on her device and can execute workflows in a web interface. Secure cloud technology provides rapid

scalability as well as data protection. Both are beneficial properties for using multimedia data in the cooking domain. With our proposed solution, the user is able to easily create cooking recipes on a platform and store multimedia data. Having started the workflow, the user is provided with cooking instructions including pictures or videos. Using secure cloud technology addresses private communities which e.g. do not want to publish content or appear in any videos accessible to the public, for instance chronically ill persons or allergy sufferers who do not want to share their special recipes or videos because their identity might be linked to serious diseases. For many people it is not an option to simply store the content on scalable platforms like YouTube because their children might be visible or because their employer might get knowledge about their special concerns. Our solution combines scalability with additional security and ensures that content is protected from persons who are not supposed to have access to the recipes. The technical solution is based on IDGARD which allows to separate data in different, sealed containers and to keep own videos or pictures confidential.

With this new solution we address the open challenge of the Computer Cooking Contest.

The remainder of our paper is structured as follows. At first we will provide an overview of our architecture and a sample of a workflow to ease comprehensibility. After that we will present our implementation concept and workflows from the cooking domain, which are based on cooking recipes of pasta. We conclude with a discussion of our research contribution and the new prospects for both companies and researchers.

2 Fundamentals

Sealed Cloud

A broad range of providers offer different models for services at different layers. But most of them do not guarantee security for the clients of a cloud provider or the privacy of the data (Santos, Gummadi et al. 2009). Especially in Germany, data protection and compliance issues require new technologies such as sealed cloud technology (Rieken 2015). A sealed cloud offers the technology to encrypt contents and meta data so that the cloud provider itself is not able to access data contents. The monitoring of user behavior and the possible access to protected content by providers is a big issue as part of data privacy and anonymity in the internet. We implement a new architecture to integrate multimedia data from the cloud using Activiti as a workflow engine and the sealed cloud IDGARD.

Copyrighted Content

From a user perspective, the access to files via the internet, for instance, by using mobile devices can be risky, because some information can be spied out by unauthorized persons. If some potential attackers retrieve user information while data is sent over the wire they may gain full access to the data storage. To prevent these attacks the usage of security tokens or passcodes (received via SMS) is necessary. IDGARD provides some of these additional security mechanisms.

From the provider perspective it is also difficult to secure data and protect it from unauthorized access or illegal sharing. Providers want to ensure that only one user or a specific user group is able to access data with the registered devices. By sending the unique keys to the registered devices, content providers can prevent the unauthorized sharing of data (e.g. by using a sealed cloud). There are similar examples in the nutrition domain where images or videos are already protected, for instance mycoachnutrition.com (MyCoachNutrition 2015). Like our approach, these services offer user individual content. In contrast to our work, they do not provide rapid scalability.

Definition of Workflows

Work is often organized as a sequence of individual tasks in which the progress can be observed (Hammer and Champy 1994). These individual tasks are linked to each other and they underlie a business objective or a policy goal (Workflow Management Coalition 1998). The automation of business processes is called workflow. According to the definition of the Workflow Management Coalition, a workflow is: "The automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules." (Workflow Management Coalition 1998). In the remainder of this paper, we will use the term "workflow" as synonym for "business process". For our project, we model different cooking instructions using Activiti as a workflow engine. A simple workflow consists of a start event, a task with a data object and an end event (see Figure 1). A task or an activity describes a piece of work that forms a logical step in a process. To support the process execution the workflow activity requires human and/or machine resources for process execution (Workflow Management Coalition 1998). We use different tasks and data objects to describe a cooking control flow.

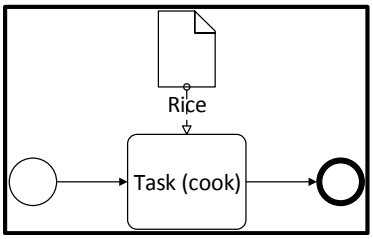


Figure 1: Sample workflow (own representation)

3 Architecture

General overview

As illustrated in Figure 2, the introduced architecture consists of three layers:

From the perspective of the user, the first layer or sub-component is based on the so called Ninja Web-Framework, which is required in order to handle the graphical interaction with the user by initiating a workflow instance.

Below the Web Framework, the Business Process Management (BPM) platform Activiti initially receives the user commands and controls the processing of the corresponding instance. The Activity engine requests relevant information from the data source (IDGARD) and hands them over to the user.

IDGARD, as the third or bottom layer of the architecture, is not just an external database but a sealed cloud solution. Therefore, it does not only store the data but provides API functions that ensure secure retrieval.

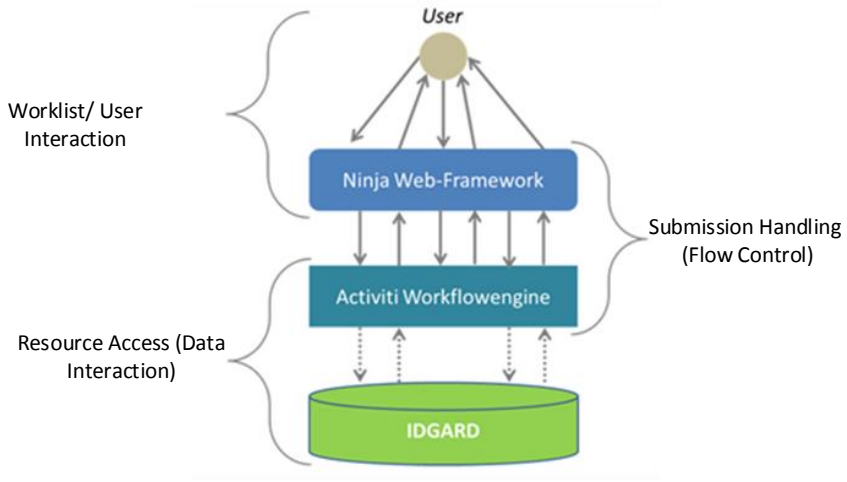


Figure 2: Architecture (own representation)

Reasons for Usage of a 3-Layer-Architecure and its Components

Since the goal is to develop and execute (cooking) workflows it is required to use Business Process Management and a solution to save the corresponding multimedia data. As a result, it is pre-determined to stage a 2-Layer-Architecure at minimum. Since it is a typical demand that users have not just local access to the workflows, a browser-based solution is reasonable. This kind of functionality is not offered by the already mentioned layers and therefore asks for an additional one.

Starting from the bottom layer, there is the complex question of how to save an application's data or where to put it. As Cloud Computing nowadays has already overcome just being a trend, it is appropriate and scientifically valuable to embed this idea into various contexts.

IDGARD´s sealed cloud is specifically designed to enhance security. Because protecting own content like graphically supported cooking steps or even whole recipes can be important, such a cloud service might be beneficial. This especially applies to companies that want to distribute such services via Internet and protect themselves against copyright infringements.

Concerning the choice of a workflow engine, there is a huge range of open source technologies available. In terms of basic functionality (e.g. integration in an IDE or graphically establishing workflows) they usually show similarities. To address our requirements properly we decided to use two different GUI's for our project. Activiti as the modelling GUI and the Ninja Web-Framework for the user worklist. However, Activiti convinces with a distinct manual, a large community and a clear, browser based testing environment.

To provide the worklist, the Ninja Web-Framework is useful because, as an integrated software stack, it already comprises many important libraries.

By using a MongoDB database, we save standardized keywords for each possible instruction and ingredient on the one hand, as well as the recipes with already assigned processing times, instructions and information on the other hand. We are planning to use time information for the retrieval of a process in our future work.

4 Implementation Concept

An abstract class in Java defines the fundamental set of functions, i.e. basic java members and abstract methods that deal with communicating information between java and JSON, and the log file for further checks and troubleshooting. Each function has still its own unique representation for requests and responses including the corresponding JSON formats. In order to successfully connect request- and response functionality for data interchange, some middleware classes are required. For providing cloud access, we have to send login-data and a random token of the client for identification purposes. The factory pattern is used to invoke requests and their corresponding response classes. The Java classes that realize the communication with the cloud server are directly implemented in the workflows, which we use for cooking instructions.

Workflow engines for Business Process Management are abounding. Examples for these are JBoss, jBPM or Activiti. For our workflow development, we use Activiti as an open source workflow engine. Activiti uses BPMN 2.0 as a modelling language and can be easily integrated with Java environments (Alfresco 2015). We used seven already created pasta cooking workflows from the work of Minor et al. and converted them into an Activiti workflow (see Figure 3 following Minor, Bergmann et al. 2010). Each cooking task is modelled by a service task. This kind of task enables us to invoke

a Java class for API cloud access. We also deposit the ingredients as data objects directly in the activity workflow. Other content like pictures or movies for cooking instructions are stored in the cloud.

In the example we have four individual instructions for the user. First, *cook* and *place in serving bowl* as well as *puree* should be conducted in parallel. Parallelization can be modelled using XOR-, AND- (symbolized by the plus), or LOOP-blocks (Schumacher, Minor et al. 2013). When both branches are finished the last task is *toss*. In each task, users get instructions with support of multimedia data from the cloud. To execute our designed workflows we use the Ninja Web framework. The Web framework (worklist) is a resource which performs the work presented by a workflow activity instance and

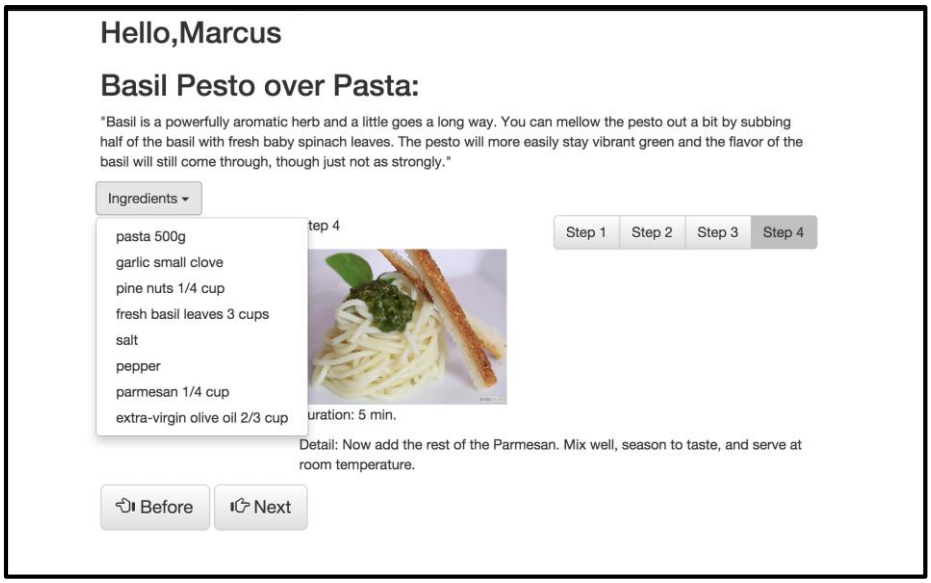


Figure 4: Current worklist (own representation)

therefore directly interacts with the user and supports her in executing her tasks (Workflow Management Coalition 1998). After successfully logging in, a user can choose to design her own recipe and upload new elements or choose ingredients to process. When searching for new processes for recipes in the database it is difficult to find a workflow or recipe which is well suited for the desired ingredients. To implement a search, case-based retrieval can be used. However, because of the small size of the recipe base in our solution this is not implemented yet.

Each recipe consists of several steps that are described by its linked ingredients, related instructions and actions that are supposed to be undertaken. In Figure 4, the prototypical implementation of our worklist is depicted. The sample picture is directly imported from the cloud. The non-multimedia information around the task is saved in a MongoDB database. When the user starts to cook she immediately sees which ingredients she needs, processing time, next steps and a video or picture to support her task. In the left top corner of Figure 4, one can see the ingredients to be used for the particular task. In the center is a management board with the instructions supported by a video or

a picture of the specific task. The workflow running in the background is modelled in BPMN 2.0. By pressing next step the user is guided to the next task of the related recipe as specified in the BPMN workflow. Thereby, we provide a worklist with different instructions for a kitchen chef.

5 Discussion and Conclusion

In our paper we present a novel approach to integrate a high security cloud storage (sealed cloud) in Business Process Management. We implement a new model for using a sealed cloud multimedia data storage for our workflow contents. The implementation of a case-based retrieval is not implemented yet. As a first step, we demonstrate a pasta-cooking workflow, which can be processed or uploaded by the user in a web form. The content is stored in the sealed cloud IDGARD of the Uniscon GmbH who provides the cloud infrastructure. The content and meta data is encrypted and protected from unauthorized provider or third-party access. By using cloud storage for our workflows we examine two main benefits: data protection and scalability. With the IDGARD solution third parties are not able to copy or even access the data.

Scalable storage provides the advantage of adapting storage up and down on-demand without using rare data space on physical disks of computers or other devices (Armbrust, Fox et al. 2009). By using IDGARD, we are able to scale our storage capacity in a flexible way in response to service usage and new customer demands. Since it is very difficult to estimate, what recipes and data with instructions will be added, scalability is very important. Apart from that, the evolution in terms of video quality has significantly risen in the past years (and probably keeps rising in the future) which contributes to the fact that scalability is an important factor because customers also tend to pressure companies to provide state-of-the-art content. The amount of data that goes along with this development is significant. It is also possible to migrate our workflow engine into a cloud solution in future. As a result, the whole platform can be scaled in response to service usage.

To sum up, our approach of using cloud storage for workflows in the cooking domain benefits from flexible scalability, higher privacy and data protection leading to a higher user experience especially for special interest groups of the cooking domain. The user no longer needs to use her limited, physical storage. She can store multimedia data with cooking instructions directly in the cloud. Our work also contributes to other future trends besides the cooking domain. Sealed cloud technology offers opportunities to use cloud storage without harming privacy or security regulations. This can be very important for audit companies who want to store audit-documents or other critical contents in the cloud.

7 Acknowledgment

We would like to acknowledge our partnership with the Uniscon GmbH who provided us with their sealed cloud service IDGARD.

8 References

Aalst, W. M., B. Benatallah and F. Casati (2003). Business Process Management, Springer-Verlag Berlin Heidelberg.

Alfresco. (2015). "Activiti mission statement." Retrieved 08.06.15, 2015, from http://activiti.org/vision.html.

Armbrust, M., A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica and M. Zaharia (2009). Above the Clouds: A Berkeley View of Cloud Computing, Technical Report, UC Berkeley Reliable Adaptive Distributed Systems Laboratory.

Ciovică, L., M. P. Cristescu and L. A. Fraţilă (2014). "Cloud Based Business Processes Orchestration." Procedia Economics and Finance **16**(0): 592-596.

Workflow Management Coalition (1998) "Terminology & Glossary." Retrieved 08.06.15, 2015, from http://www.wfmc.org/resources.

Hammer, M. and J. Champy (1994). Business Reengineering: Die Radikalkur für das Unternehmen. Campus-Verlag Frankfurt.

Minor, M., R. Bergmann, S. Görg and K. Walter (2010). Adaptation of cooking instructions following the workflow paradigm, In: ICCBR 2010 Workshop Proceedings, pp. 199-208.

MyCoachNutrition. (2015). "MyCoachNutrition." Retrieved 06.07.15, 2015, from http://www.mycoachnutrition.com/.

Rieken, R. (2015). Datensicherheit als Herausforderung im Cloud-Computing-Trend. In: Marktplätze im Umbruch**:** pp 751-759, Springer.

Santos, N., K. P. Gummadi and R. Rodrigues (2009). Towards trusted cloud computing. In: Proceedings of the 2009 conference on Hot topics in cloud computing. San Diego, California, USENIX Association.

Schulte, S., C. Janiesch, S. Venugopal, I. Weber and P. Hoenisch (2015). "Elastic Business Process Management: State of the art and open challenges for BPM in the cloud." Future Generation Computer Systems **46**(0): 36-50.

Schumacher, P., M. Minor and E. Schulte-Zurhausen (2013). Extracting and enriching workflows from text. In: Proceedings of the 2013 IEEE 14th International Conference on Information Reuse and Integration (IRI), pp. 285-292.

Wang, J., P. Korambath, I. Altintas, J. Davis and D. Crawl (2014). "Workflow as a Service in the Cloud: Architecture and Scheduling Algorithms." Procedia Computer Science **29**(0): 546-556.

The Doctoral Consortium

At the
Twenty-Third International Conference on
Case-Based Reasoning
(ICCBR 2015)

Frankfurt, Germany
September 2015

Nirmalie Wiratunga and Sarah Jane Delany (Editors)

Program Chairs

Nirmalie Wiratunga	The Robert Gordon University, Scotland
Sarah Jane Delany	Dublin Institute of Technology, Ireland

Program Committee

David W. Aha	Naval Research Laboratory, USA
Klaus-Dieter Althoff	DFKI / University of Hildesheim, Germany
Kerstin Bach	NTNU, Norway
Amélie Cordier	LIRIS, France
Susan Craw	The Robert Gordon University, Scotland
Odd Erik Gundersen	NTNU , Norway
Joseph Kendall-Morwick	Capital University, USA
David Leake	Indiana University, USA
Jean Lieber	LORIA - INRIA Lorraine, France
Cindy Marling	Ohio University, USA
Stewart Massie	The Robert Gordon University, Scotland
Stefania Montani	University Piemonte Orientale, Italy
Miltos Petridis	University of Brighton, UK
Barry Smyth	University College Dublin, Ireland
David Wilson	University of North Carolina at Charlotte, USA

Preface

The Doctoral Consortium volume contains the research summaries that were presented at the 7th Annual ICCBR 2015 Doctoral Consortium (www.iccbr15.de/index.php/2014-10-27-10-03-12/doctoral-consortium) held on 28th Sep 2015 in Frankfurt, Germany. There were 11 accepted submissions consisting of: (1) a 1-page Application Cover Page; (2) a 3-page Research Summary; (3) a 1-page Resum; and (4) a letter of support from the student's advisor. The objectives, progress, plans and references in each research summary have been progressively refined according to feedback from one or more PC members, of which one was also the assigned mentor. Feedback was organised into three broad areas: general outlook in terms of research hypothesis and proposed methodology; detailed comments specific to the student's project; and finally advice for the talk presentation. A face-to-face pre-event meeting opportunity enabled all student-mentor pairs to refine their presentations. Mentors also had the responsibility of leading the question and answer session following each mentee presentations on the day.

The ICCBR-15 DC began on September 27th with an informal meet and greet session, followed by a discussion led by Dr Kerstin Bach (Norwegian University of Science and Technology) on shared student experiences. The evening ended with dinner sponsored by the conference. On September 28th, the formal program started with an invited talk by Prof Klaus-Dieter Althoff (Hildesheim University), entitled Lessons Learned - A Journey from Research Student to Professor. The rest of the program consisted of 15-minute talks presented by 11 doctoral students on their Research Summary. The presentations covered a wide range of CBR topics including recommender systems, retrieval, adaptation and maintenance in CBR, e-learning systems, agents and analogical reasoning, distributed CBR and AI in music.

Many people participated in making the DC event a success. We wish to thank all our PC members who provided important and useful guidance to DC students, either as reviewers or as mentors. We are very grateful for the generous support of the sponsors of the ICCBR-15 DC: The AI Journal and National Science Foundation. Once again AIJ has enabled us to provide significantly discounted registration fees to our participants and the NSF funding obtained through David Wilson has helped fund travel costs for our students from the US. Finally thanks go out to David W. Aha who has helped muster a healthy number of participants for this year's event.

Finally thank you to all our DC participants. We trust that the ICCBR-15 DC enhanced your interest in studying CBR and that the welcome and support from the CBR community has sparked your interest in this field for many years to come.

September 2015 — Nirmalie Wiratunga
Frankfurt — Sarah Jane Delany

Distributed Case-based Support for the Architectural Conceptualization Phase

Viktor Ayzenshtadt

University of Hildesheim, Institute of Computer Science
Competence Center Case-Based Reasoning
German Research Center for Artificial Intelligence
Trippstadter Straße 122, 67663 Kaiserslautern, Germany
ayzensht@uni-hildesheim.de

1 Introduction

When an architect conceptualizes a new building she is very likely in need of new ideas, solutions and inspiration to create a new design. *Metis* [3] is a basic case-based design research project of the German Research Center for Artificial Intelligence (DFKI) and the KSD Research Group of the TU Munich that aims to help architects during the early design concept stage and corresponding building plans creation by providing them with similar building designs to a created one. One of the main aspects of the project is the creation of new cases (building designs) by transforming floorplan sketches with image processing techniques into graph representations which are based on the *Semantic Fingerprint* [4] model. Another one is the retrieval process that uses a multi-agent system with case-based agents that are able to apply either subgraph matching or CBR-framework-based retrieval to find similar building designs. An architect can search for them by using a web browser-based graphical interface. As usual, the project also includes participation of experts, who discuss and explain the details and aspects of the CAAD, CBR and Multi-agent systems research tasks.

In my master thesis I extended the previously existing initial concept of the retrieval system to provide the core functionality for the project's retrieval tasks. This system uses the retrieval container structure where each container acts as a separate multi-agent system that is only responsible for resolving a single user query. The retrieval process is coordinated by a corresponding agent. The case base consists of extracted and imported graph representations of the building designs. The gateway supports the connection between the core systems and the user interface.

2 PhD Research Focus

In my PhD thesis research I am going to concentrate on the research fields named in Section 1 and continue to study the case-based architectural design support questions. The implemented retrieval system from the master thesis will be taken as a base and extended for the further research. In detail, the currently planned research goals are described in the following sections.

2.1 Case Representation

This research part will answer the question which model is the most preferable one for representing architectural design cases in CBD applications – graphs or *attribute-value* concepts. The comparison of those models will include the study on how both of them perform under the same conditions when conducting retrieval and inserting of new cases. Currently cases consist of graph-based, GraphML-based [1], myCBR-based [2] and ontologically applied multi-agent communication language FIPA-SL-based floorplan representations that include room representations and room connections with corresponding attributes and values. The knowledge for creating those cases is acquired and maintained by the specific maintainer system agent that obtains, transforms, separates and inserts building design graphs into the corresponding case bases.

2.2 Retrieval Performance

The cross-validation of both retrieval approaches – *subgraph matching* and *CBR-framework-based* – is another part of the planned research. Here both approaches will be validated by applying the cross-comparison between those two types. The aim of this process is to answer the question which of both approaches provides the highest quality of the retrieval results. Both retrieval models will be confronted with different user scenarios to find the best suitable method for a given situation or context.

2.3 Retrieval Coordination

Two currently available retrieval coordination approaches – *rule-based* and *case-based* – are going to be extended to a full functionality and provide a complete pool of features needed for the relevant query. In addition a cross-comparison of them as a part of the retrieval performance measurement could be performed as well. Architectural experts' help and users' feedback can be taken into account and used for the evaluation of the result quality.

2.4 CBR Domain Modelling

The myCBR part of the retrieval system contains the CBR domain that is based on the structure of the *Semantic Fingerprint* model. The underlying model of the domain is going to be improved (with the experts' help inter alia) and adapted to the results of the studies named in the previous research goals.

This aim is also valid for the CBR agents, the retrieval system entities that are responsible for the last step of the retrieval of the similar building designs. The case-based learning feature of those agents implements an own CBR domain component. This component is unique for each of the currently existing CBR agent types. It provides the corresponding agent with the reasoning functionality in order to support its decision when it comes to select the proper retrieval strategy and similarity measures.

2.5 Applying the Generic Framework Beyond Architectural Design

From the above described multi-agent-system-supported CBR-based retrieval a generic framework will be developed and applied to other domains than architecture. One specific focus will be under which constraints the generic framework can help to overcome the inherent complexity of searching for optimal subgraphs. Based on the results an according domain and task characterization will be developed. Other research focus will be dealing with the generalization of the learning agents approach for CBR-based information retrieval for design ideas generating process. The goal is to formalize and optimize the agents' experience and knowledge obtaining, teamwork and communication process in order to provide an efficient distributed case-based IR approach that is able to find information with high precision and recall rates in one or more case bases with differently (e.g. only partly) structured knowledge representation types and domain models. Consideration of applying similarity or diversity as the best suitable case comparison base will also be taken into account and a part of agents' reasoning process.

3 Current Progress

The current progress state is now in the initial phase. The research group of *Metis* is currently evaluating the user interface for creating the user queries in AGraphML (*Architectural* GraphML) format. The next steps are the integration of the interface into the retrieval system and the implementation of subgraph matching algorithms to be able to use them as second possible retrieval approach.

In the following research phase it is planned to find an explicit research direction of the PhD thesis, that can be either one of the described research foci or a combination of some of them with or without adding some new aspects that can appear during the ongoing *Metis* project discussions.

References

1. Brandes, U., Eiglsperger, M., Herman, I., Himsolt, M., Marshall, M.S.: Graphml progress report structural layer proposal. In: Graph Drawing. pp. 501–512. Springer (2002)
2. DFKI GmbH: myCBR 3 tutorial information for ICCBR 2012 (2012), http://mycbr-project.net/downloads/myCBR_3_tutorial_slides.pdf
3. KSD Research Group: KSD - Forschungsprojekte - metis (2015), http://ksd.ai.ar.tum.de/?page_id=140
4. Langenhan, C., Petzold, F.: The fingerprint of architecture-sketch-based design methods for researching building layouts through the semantic fingerprinting of floor plans. International electronic scientific-educational journal: Architecture and Modern Information Technologies 4, 13 (2010)

All links were last followed on July 8, 2015.

Interactively Learning Moral Norms via Analogy

Joseph Blass

Ph.D. Candidate, Qualitative Reasoning Group, Northwestern University, Evanston, IL
joeblass@u.northwestern.edu

Abstract. Autonomous systems must consider the moral ramifications of their actions. Moral norms vary among people, posing a challenge for encoding them explicitly in a system. This paper proposes to enable autonomous agents to use analogical reasoning techniques to interactively learn an individual's morals.

1 Introduction

1.1 Challenge and Research Goals

Should a self-driving car put its passengers at risk and swerve to avoid a jaywalker, or protect its passengers and hit him? To participate in our society, computers need to share our ethics. As these systems become more autonomous, they must consider the moral ramifications of their actions. I intend to build an AI moral-reasoning system that strives for good, but can select amongst only bad options, by acquiring and applying human morals. This system will learn moral norms through natural-language interaction with humans and analogical generalization, and apply these norms by analogy.

The diversity of moral norms and concerns make hand-encoding an individual's moral sense or providing case-by-case instructions impossible. Natural interaction will be key, since users may have neither the technical skills nor understand their own morals enough to encode them themselves. Also, since human morals likely do not depend on first-principles reasoning (FPR) (Haidt, 2001), and since moral rules contradict and trade off with each other, I intend to minimize FPR in the system. A pure FPR moral reasoning system would either need rules for all possible trade-offs, to be able to ignore certain morals (a bad idea), or would freeze when moral obligations conflict. Analogical reasoning can avoid these problems if provided a good analogue.

1.2 MoralDM, Structure-Mapping, and the Companions Architecture

MoralDM (Dehghani et al. 2009) is a computer model of moral reasoning that takes in a moral dilemma in natural language, uses a natural language understanding (NLU) system to generate Research-Cyc-derived predicate-logic representations of the dilemma, and uses analogy over resolved cases and FPR over explicit moral rules to make moral decisions consistent with humans'. MoralDM is the starting point for my work.

The Structure Mapping Engine (SME), based on Gentner's (1983) Structure Mapping Theory of analogy, constructs an alignment between two relational cases and

draws inferences from it. SME can apply norms by analogy from stories (Dehghani et al. 2009). Analogy is a good fit for moral decision-making because both are guided by structure, not features. Consider the following examples. 1) A bomb will kill nine people in a room, but you can toss it outside, where it will kill one person. 2) A bomb will kill nine people, but you can toss someone onto it to absorb the blast and save the nine. Most say tossing the bomb, but not the person, is morally acceptable. These scenarios only differ structurally, in what fills which role; the entities and action types themselves are shared. The classic trolley problem (a trolley will hit five people unless it is diverted to a side track where it will hit one person), in contrast, has different features, but the same structure, as the first bomb case. Humans see these two cases as morally alike.

The Sequential Analogical Generalization Engine (SAGE) builds case generalizations that emphasize shared, and deprecate case-specific, structures. SAGE uses a case library of generalizations and exemplars. Generalizations contain facts from constituent cases: non-identical corresponding entities are replaced by abstract ones; probabilities indicate the proportion of assimilated cases each fact is present in. Given a probe, SAGE uses SME to find the most similar case in its case library. If the match is strong enough, the case is assimilated; if not, it is added as an exemplar. SAGE can use near-misses to determine defining characteristics of category members (McLure et al., 2015).

The Companion Cognitive Architecture emphasizes the ubiquity of qualitative representations and analogical reasoning in human cognition. Companion systems are designed to work alongside and interactively with humans (Forbus & Hinrichs, 2006).

2 Proposed Research and Progress

I propose to extend MoralDM in the Companion Architecture to learn to model a human user's morals. The system will learn to recognize and extract moral norms through the generalization process. It will get moral stories in natural language from the user, generate qualitative representations of those stories, generalize over those representations, and use SME to apply morals from the generalizations. I will extend MoralDM's analogical reasoning, integrate emotional appraisal, and improve NLU for a moral lexicon.

Previously MoralDM's analogical reasoning module exhaustively matched over resolved cases, which is computationally expensive and cognitively implausible. SME over ungeneralized cases also sees feature-similar but morally-different cases (i.e., the bomb scenarios) as a good match, due to the amount they have in common.

MAC/FAC is a two-step model of analogical retrieval. MAC efficiently computes dot-products between the content vectors of the probe and each case in memory (a coarse similarity measure). FAC then performs SME mappings on the most similar cases. MAC sees cases concerning mostly the same entities as the probe as good potential matches, even if the structures differ. Using MAC/FAC over generalizations rather than exemplars solves this problem, since generalizations emphasize defining structure. Abstract generalizations applied by analogy can therefore function as moral rules.

We have found that reasoning by analogy over generalizations led to more human-like judgments than using ungeneralized cases (Blass & Forbus, 2015). Reasoning can be further improved using McLure & Forbus' (2015) work on near-misses to illustrate category boundaries and the conditions for membership or exclusion. MoralDM also

still reasons using FPR about facts relevant to moral judgment, such as directness of harm. These are not explicitly stated, though we recognize them easily; MoralDM uses them in a consistency check to ensure the quality of retrieved analogues. Near-misses would let MoralDM use analogy, not FPR, to find the facts for the consistency check.

We want to expand the range and provenance of stories for MoralDM to learn from. One option is to crowd-source moral stories to present to a user for endorsement or rejection, rather than force the user to provide them all. QRG's NLU system, EA NLU, generates qualitative representations from English input, but its moral vocabulary is currently limited. The Moral Foundations Dictionary (Graham et al., 2009) is a moral lexicon; to enable EA NLU to understand moral stories, I will ensure lexical and ontological support for this vocabulary. Another NLU challenge is how to infer information implicit in the text. Work has been done at QRG on inferring narrative information, including about moral responsibility (Tomai & Forbus, 2008). I will extend EA NLU's abductive reasoning as needed to support moral narrative understanding. Finally, I will integrate emotional appraisal (Wilson et al. 2013) into MoralDM. Emotional appraisal can help recognize moral violations and enforce moral decisions.

My goal is to have a Companion running MoralDM with the above extensions interact with a human and build a model of their moral system. MoralDM could not previously do this, since it required all moral norms to be explicitly encoded, and modeled a society's aggregate judgments, not individuals. The new system will have the human tell it a moral story, crowd-source thematically similar stories, and ask the human which illustrate the same moral principle (the others are near-misses). For each story, the system would predict the moral value of actions and compare its predictions to the human's moral labels. When the core facts of the generalization stop changing and the system's labels consistently match the human's, the system has mastered that moral domain.

This project brings challenges. How much FPR will remain necessary? How must EA NLU be extended to understand moral narratives? What narrative inferences should be made about implicit information? Nonetheless, I believe I can build a system that interactively learns to model an individual's morality.

3 References

Blass, J. & Forbus, K. (2015). Moral Decision-Making by Analogy: Generalizations vs. Exemplars. *Proceedings of the 29th AAAI Conference on Artificial Intelligence*, Austin, TX.

Dehghani, M., Sachdeva, S., Ekhtiari, H., Gentner, D., & Forbus, K. (2009). The role of cultural narratives in moral decision making. In *Proceedings of the 31st Annual Conference of the Cognitive Science Society*.

Forbus, K., & Hinrichs, T. (2006). Companion Cognitive Systems: A Step Towards Human-Level AI. *AI Magazine* 27(2), 83-95.

Gentner, D. (1983). Structure-Mapping: A Theoretical Framework for Analogy. *Cognitive Science* 7(2).

Graham, J., Haidt, J., & Nosek, B. A. (2009). Liberals and conservatives rely on different sets of moral foundations. *Journal of personality and social psychology*, *96*(5), 1029.

Haidt, J. (2001). The Emotional Dog and its Rational Tail: A Social Intuitionist Approach to Moral Judgment. *Psychological Review*, 108(4), 814-834.

McLure, M.D., Friedman S.E. and Forbus, K.D. (2015). Extending Analogical Generalization with Near-Misses. In *Procs of the 29th AAAI Conference on Artificial Intelligence*, Austin, TX

Tomai, E. & Forbus, K. (2008). Using Qualitative Reasoning for the Attribution of Moral Responsibility. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society*. Washington, D.C.

Wilson, J. R., Forbus, K. D., & McLure, M. D. (2013). Am I Really Scared? A Multi-phase Computational Model of Emotions. In *Proceedings of the Second Annual Conference on Advances in Cognitive Systems.*

Support to Continuous Improvement Process in manufacturing plants of multinational companies through Problem Solving methods and Case-Based Reasoning integrated within a Product Lifecycle Management infrastructure

Alvaro Camarillo

Mechanical Engineering Department, Universidad Politécnica de Madrid
a.camarillo@alumnos.upm.es

1 Introduction to problems addressed by research

The aim of this research is to capture and reuse efficiently knowledge at shop floor level of multinational companies during the resolution of manufacturing daily problems (e.g. scrap rate, quality issues, breakdowns, and in general any Continuous Improvement Process (CIP) activity). We want to provide production technicians and operators with a friendly and low time consuming Knowledge Management (KM) tool to get their engagement and collaboration, avoiding negative impact in productivity, and promoting the knowledge share across plants overtaking language, nationalities, and competition barriers.

We propose the Problem Solving (PS) method 8D as structured process to guide the knowledge share. A Product Lifecycle Management (PLM) system will be the logical infrastructure to store all product, process, machinery, and users information. This PLM system will host also the database of a Case-Based Reasoning (CBR) system. This CBR system will be the KM tool in charge of capturing and reusing the knowledge [1],[4],[5],[6]. FMEAs of Design, Process and Machinery will be used to populate initially the CBR System [3].

The CBR cycle [1] would be though as follow:

- User introduces basic description of de problem (new case).
- Based on this description the CBR system collects additional information related product, process, machinery or users from the PLM. It proceeds to calculate to find similar cases.
- The system proposes containment actions and different root causes (retrieved cases). The user checks these root causes in the line and gives feedback to system.
- Based on the corrected list of most similar cases the system performs adaptation and proposes a solution (solved case).
- Solution is tested by user (tested/repaired case) and implemented together with its associated preventive actions.

– The learned case is stored in the database of cases.

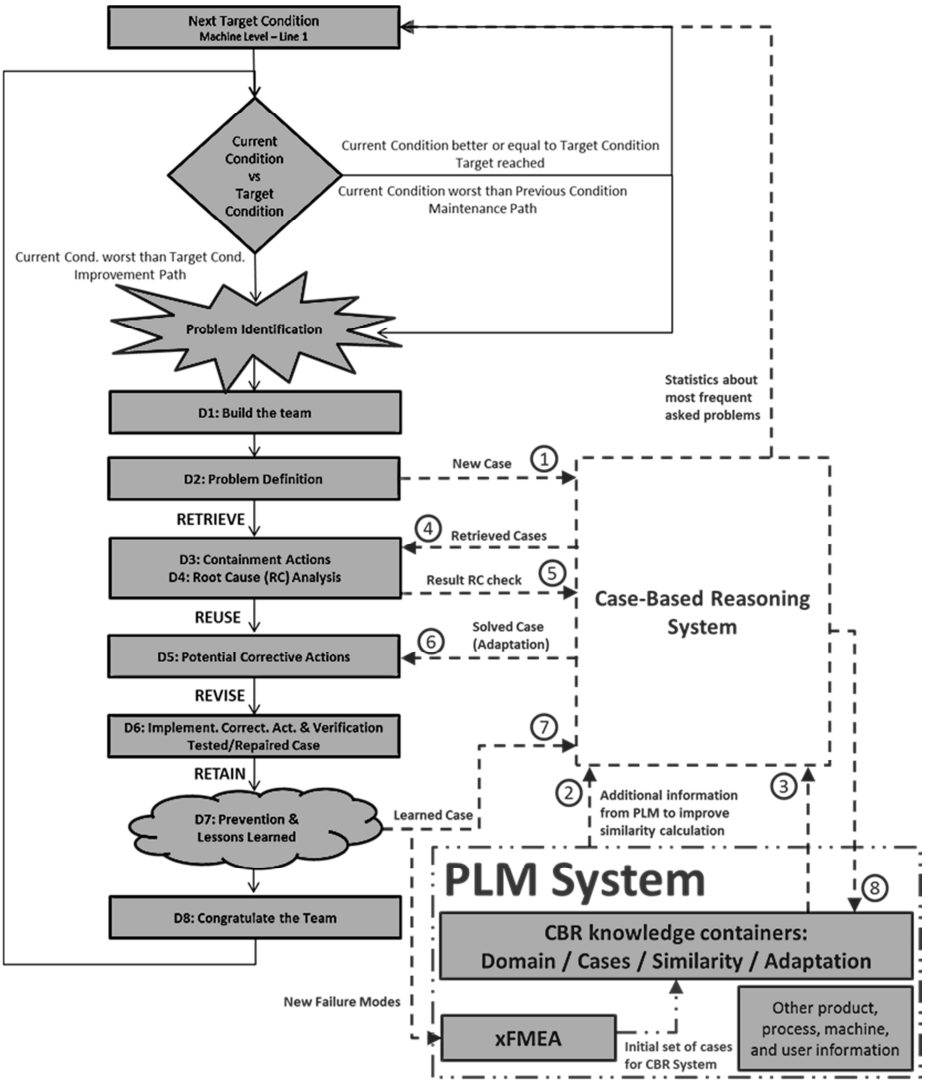


Fig. 1. – Proposed model for supporting CIP through PS, CBR and PLM

Expected contributions of this research:

– Combination in a single model of PS methods, as process for guiding the share of knowledge, CBR, as KM tool, PLM as global infrastructure to contain information and control information flow, and FMEA method, as the tool to define manufacturing problems in a formalized way and to populate initially the CRB System.

- Based on minimal data introduced by the user (low time consumed) we propose to get from the PLM extended information that will be used to calculate similarity.
- Bring this type of KM tool not only to designers or to engineers, but direct to blue-collar associates working at production lines.

2 Description of progress to date

Currently we are developing the Model that has to support the knowledge capture and reuse (see fig. 1). For the case study, two open source applications have been selected: Aras as PLM software (www.aras.com), and myCBR as CBR software (www.mycbr-project.net) [2]. For the implementation, a multinational company of the electrical batteries branch was selected. To get the benefits of knowledge sharing between two teams with very low interaction until now the system will be installed in two manufacturing plants located in two different countries. It will focus only on one of the production steps of batteries in order to get consistent results in a limited period of time.

3 Proposed plan for research

After the review of the state of the art in the fields of CIP, PS, CBR, and PLM, we are currently designing the initial knowledge containers of domain, similarity and adaptation of the CBR system [6] that will be used to test our concept initially in a single production line. This task has to be finished by the end of May 2016. The experience from this initial test will be used to improve our KM tool in order to do a second test loop at whole plant level. Finally a third test loop will be performed between the two plants until end of 2016. The presentation of the PhD Thesis is planned in May 2017.

References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. In: AI Communications, vol. 7, no. 1, pp. 39-59 (1994)
2. Atanassov, A., Antonov, L.: Comparative Analysis of Case Based Reasoning Software Frameworks jCOLIBRI and myCBR. In: Journal of the University of Chemical Technology and Metallurgy, vol. 47, no. 1, pp. 83-90 (2012)
3. Atamer, A.: Comparison of FMEA and field-experience for a turbofan engine with application to case based reasoning. In: Aerospace Conference, 2004. Proceedings. 2004 IEEE, Vol. 5, pp. 3354-3360 (2014)
4. Jabrouni, H., Kamsu-Foguem, B., Geneste, L., Vaysse, C.: Continuous Improvement Through Knowledge-Guided Analysis in Experience Feedback. In: Engineering Applications of Artificial Intelligence, vol. 24, no. 8, pp. 1419-1431 (2011)
5. Kamsu-Foguem, B., Couderta, T., Bélera, C., Genestea, L.: Knowledge Formalization in Experience Feedback Processes: An Ontology-Based Approach. In: Computers in Industry, vol. 59, no. 7, pp. 694-710 (2008)
6. Richter, M., Weber, R.: Case-Based Reasoning: A Textbook. Springer, Heidelberg (2013)

Aspect-based Sentiment Analysis for Social Recommender Systems

Yoke Yie Chen

IDEAS Research Institute,
Robert Gordon University,
Aberdeen, Scotland
{y.y.chen}@rgu.ac.uk

1 Introduction

Social recommender systems provide users with a list of recommended items by exploiting knowledge from social content. Representation, similarity and ranking algorithms from the Case-Based Reasoning (CBR) community have naturally made a significant contribution to social recommender systems research [1, 2]. Recent works in social recommender systems have been focused on learning implicit preferences of users from online consumer reviews. Most online reviews contain user opinion in the form of positive and negative sentiment on multiple aspects of the product. Since a product may have multiple aspects, we hypothesize that users purchase choices are based on comparison of products; which implicitly or explicitly involves comparison of aspects of these products. Therefore, our main research question is "*Does considering product aspects importance (weight) improve prediction accuracy of a product ranking algorithm*?"

2 Research Aim

This research aims to develop a novel aspect-based sentiment scoring algorithm for social recommender systems. Our particular focus will be on using social content to develop novel algorithms for different product domains. For this purpose, we intend to:

1. Develop an aspect extraction algorithm to extract product aspects.
2. Develop aspect weighting algorithms to extract product aspects weights from social content.
3. Study the effect of temporal dynamics on aspect weight.
4. Evaluate the performance of our proposed algorithms in performing a top-N recommendation task using standard performance metrics such as mean average precision.

3 Challenges

Social recommender system harness knowledge from product reviews to generate better recommendation. Key to this task is the need for a novel aspect based sentiment analysis approach to harness this large volume of information. However, this approach suffers three main challenges:

1. Aspects extracted from product reviews using NLP-based techniques rely on POS tagging and syntactic parsing which are known to be less robust when applied to informal text. As a result, it is not unusual to have a large numbers of spurious content to be extracted incorrectly as aspects.
2. A user's purchase decision hints at the aspects that are likely to have influenced their decision and as such be deemed more important. To understand the importance of an aspect to users, it is necessary to further reveal the importance weight that users placed on an aspect. Additionally, user preferences change over time. Term frequency (TF) is the naive approach for this task where the weight of an aspect is equal to the number of occurrences of that aspect in product reviews. However, this approach is not able to capture users' preferences that change over time.
3. The absence of ground truth data causes evaluating ranking algorithm a challenging task in recommender system. For example, Best Seller ranking in Amazon can be a straightforward reference to evaluate the ranking of system generated recommendation list. However, this ranking is biased towards old products in Amazon. Therefore, there is a need to study relevant knowledge sources to construct a reference ranking for evaluation purpose.

4 Proposed Plan of Research

To answer our research question, our proposed plan of research is:

1. Compare the performance of our proposed aspect extraction algorithm with key state-of-the-art algorithms to determine the impact of aspect quality on recommendation tasks. We will evaluate the performance of these algorithms through accuracy metrics in extracting genuine product aspects. Thereafter, we apply the extracted aspects from these approaches in our aspect based sentiment scoring algorithm and rank the products. We then compare the recommendation performance of aspect based sentiment scoring algorithm with a sentiment analysis algorithm that is agnostic of aspects.
2. Feature selection techniques in machine learning are known to enhance accuracy in supervised learning tasks such as text classification by identifying redundant and irrelevant features. We propose to explore different feature selection techniques (e.g. Information Gain and Chi-squared) to select aspects that are important to users.
3. Our initial approach in aspect weighting algorithm places individual product aspect with equal importance weight across all products. We intend to

explore other related approaches such as TF-IDF (Term Frequency Inverse Document Frequency) to represent the importance of a product aspect. TF-IDF has been widely used in Information Retrieval community to evaluate the importance of a word to a document in a corpus. We propose to augment our aspect weighting algorithm by evaluating the importance of a product aspect to a particular product.

4. To study the effect of temporal dynamics in aspect importance weight, we look into investigating aspect weights that are inferred by:
   - **Trending information**. We would like to analyse different trending patterns of aspects occurrence in product reviews over the years (e.g. upward, downward and recurring trend). Specifically, a higher weight should be given to aspects which have an upward and recurring trend, indicating that the importance of an aspect is growing. Likewise, a lower weight should be given to aspects having a downward trend.
   - **Recency of aspects**. Aspects which frequently appear in old product reviews will have a lower weight than aspects appearing in recent product reviews. This indicates that aspects that are frequently occurring in recent product reviews are deemed important.
5. To evaluate our ranking algorithm, we use users' ratings as the baseline to compare with our proposed ranking approach. This baseline ranks each product using the average users' rating. Products in the higher rank are thus recommended.

5 Current Progress

Designed and developed novel algorithms in the following areas:

- **Aspect extraction**. The proposed approach integrates semantic relationship and frequency cut-off. The proposed approach was evaluated against state-of-the-art techniques and obtained positive results.
- **Aspect selection**. We address the problem of selecting important aspects using feature selection heuristics based on frequency counts and Information Gain (IG) to rank and select the most useful aspects.
- **Aspect-based sentiment scoring**. The proposed algorithm incorporates aspect importance weight and sentiment distribution. We investigated two different resources that infer the importance of product aspects: preference and time.

References

1. R. Dong, M. Schaal, M. OMahony, K. McCarthy, and B. Smyth. Opinionated product recommendation. In *Inter. Conf. on Case-Based Reasoning*. 2013.
2. L. Quijano-Sánchez, D. Bridge, B. Díaz-Agudo, and J. Recio-García. Case-based aggregation of preferences for group recommenders. In *Case-Based Reasoning Research and Development*, pages 327–341. 2012.

Toward a Case-Based Framework for Imitation Learning in Robotic Agents

Tesca Fitzgerald

School of Interactive Computing,
Georgia Institute of Technology,
Atlanta, Georgia, 30332
tesca.fitzgerald@cc.gatech.edu

1 Introduction

Imitation learning is a skill essential to human development and cognition [6, 5]. Naturally, imitation learning has become a topic of focus for robotics research as well, particularly in interactive robots [1, 2]. In imitating the actions of a teacher, a cognitive agent learns the demonstrated action such that it may perform a similar action later and achieve a similar goal. Thus, we expect that a cognitive robot that learns from imitation would reuse what it has learned from one experience to reason about addressing related, but different, problem scenarios.

The eventual goal of this work is to use a case-based approach to enable imitation learning in interactions such as the following. A human teacher guides the robot to complete a task, such as scooping the contents of one container into another. The robot records the demonstrated actions and observed objects, saving the demonstration as a *source case* in its case memory. At a later time, the robot is asked to repeat the *scooping* task, but in a new, *target* environment containing a different set of object features to parameterize and execute the task. Next, the robot would transfer its representation of the *scooping* task to accommodate for the differences between the source and target environments, and then execute an action based on the transferred representation to achieve the goal state in the target environment.

Using a case-based framework to address this problem allows us to represent demonstrations as individual experiences in the robot's case memory, and provides us with a framework for identifying, transferring, and executing a relevant source case demonstration in an unfamiliar target environment. The main research questions we plan to address are as follows:

- How should task demonstrations be represented in case memory?
- How do we determine which features of a robot's environment are relevant to completing a task, and thus should be stored in the source case?
- What features should be considered in retrieving a source case demonstration for reuse in a target environment? How should these features be prioritized during source case retrieval?

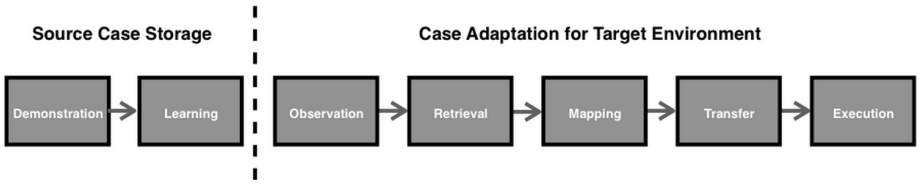


Fig. 1. Case-Based Process for Task Demonstration Transfer

2 Research Plan and Progress

We have defined a case-based approach to transfer for enabling imitation in robotic agents, consisting of two separate processes (as shown in Figure 1): the *Case Storage* process in which the robot receives demonstrations of a task and stores each demonstration as a case in source memory, and a *Case Adaptation* process which is used at a later time when the robot is asked to repeat a task in a target environment.

2.1 Case Storage Process

Demonstration and Learning We have implemented the first step in the *Case Storage* process, where the robot records and stores each task demonstration as a source case in memory. We define each case as the tuple $C = <L,\ D,\ T,\ O,\ S_i,\ S_f>$, where:

- L represents the label of the task which was demonstrated, e.g. "scooping".
- D represents the set of action models which encode the demonstrated motion, represented as Dynamic Movement Primitives as defined in [4].
- T is the set of parameterization functions which relate the set of action models to the locations of objects in the robot's environment. For example, a parameterization function may be used to represent how the robot's hand must be located above a bowl prior to completing a *pouring* action.
- O is the set of *salient* object IDs which are relevant to the task.
- S_i and S_f are the initial and final states, respectively, which represent the set of objects observed in an overhead view of the robot's environment.

2.2 Case Adaptation Process

At a later time, the robot may be asked to repeat a learned task in an unfamiliar target environment. Using the framework shown in Figure 1, the robot may address a target environment using the following steps.

Observation The robot is given a target problem to address, under the assumption that it has a relevant source case in memory which can be used to address the target problem. The robot observes the target environment by viewing the objects located in the table-top environment using an overhead camera, providing it with the initial state S_i of the target case.

Retrieval and Mapping The robot must then choose a source case from memory containing the demonstration that is most relevant to the current target problem. Once a relevant source case has been retrieved, a mapping must be generated that encodes the differences between the source and target environments. This mapping is later used to transfer the source case such that differences in the target environment are addressed. We have not yet implemented the Retrieval and Mapping steps, but will be addressing them in upcoming work.

Transfer and Execution We have implemented the last two steps of the Case Adaptation process, the *Transfer* and *Execution* steps. Currently, we manually provide the robot with the most relevant source case demonstration and a mapping between objects in the source and target environments.

We take a similarity-based approach to transfer, where we consider the similarity between the source case and target environments when defining transfer processes. As we encounter transfer problems in which the source and target problems become less similar, the source case is transferred at a different level of abstraction, such that only high-level features of that case are transferred. We have implemented three transfer methods, each of which operates by transferring the source case at a different level of abstraction (further described in [3]). Once the source case has been transferred, it is used to plan and execute a new action trajectory to address the target problem. Preliminary experiments have evaluated each method under the assumption that we select the approach, and thus the level of abstraction at which transfer occurs, to be used for a given transfer problem.

3 Future Work

Our current implementation assumes that we manually provide a mapping between equivalent objects in the source and target environments. We plan to identify (i) a method for autonomously determining this object mapping and (ii) a process for identifying and retrieving an appropriate source case demonstration.

References

1. Argall, B.D., Chernova, S., Veloso, M., Browning, B.: A survey of robot learning from demonstration. Robotics and Autonomous Systems 57(5), 469–483 (2009)
2. Chernova, S., Thomaz, A.L.: Robot learning from human teachers. Synthesis Lectures on Artificial Intelligence and Machine Learning 8(3), 1–121 (2014)
3. Fitzgerald, T., Goel, A.K., Thomaz, A.L.: A similarity-based approach to skill transfer. Women in Robotics Workshop at Robotics: Science and Systems (2015)
4. Pastor, P., Hoffmann, H., Asfour, T., Schaal, S.: Learning and generalization of motor skills by learning from demonstration. In: Robotics and Automation, 2009. ICRA'09. IEEE International Conference on. pp. 763–768. IEEE (2009)
5. Piaget, J., Cook, M.T.: The origins of intelligence in children. (1952)
6. Tomasello, M., Kruger, A.C., Ratner, H.H.: Cultural learning. Behavioral and brain sciences 16(03), 495–511 (1993)

Doctoral Consortium Research Summary: Virtuosity in Computational Performance

Callum Goddard

School of Electronic Engineering and Computer Science,
Queen Mary University of London, Mile End Road,
London E1 4NS, United Kingdom
c.goddard@qmul.ac.uk
http://www.eecs.qmul.ac.uk/

Abstract. This is a research summary of Virtuosity in Computational Performance, addressing the question: *How can a computer, as judged by a human audience, demonstrate virtuosity in computational performances with a physical model of a bass guitar?* The proposed plan for this research is to develop a computational performance system which uses case-based reasoning and reflection to produce virtuosic performances with a physical model of an electric bass guitar. Three supporting studies are planned to investigate bass playing, collect performance data and perceptions of virtuosity.

Keywords: Computational Performance, Virtuosity, Case-based Reasoning, Reflection, Physical Modelling, Music Analysis

1 The Problem being Addressed and Research Questions

The main question this research is addressing is:

> *How can a computer, as judged by a human audience, demonstrate virtuosity in computational performances with a physical model of a bass guitar?*

Computationally performed music, where a computer is responsible for rendering, generating or synthesising the music in its entirety, can appear lacking, robotic or sterile [1]. There are approaches to overcome this that focus on introducing or emulating expressive phrasing within a performance of a score [1]. However, if instead of expression virtuosity was exhibited within a computer performance, would this not offer a more satisfying solution to sterile performances as well as aiding in investigations into virtuosity of human performances?

Virtuosity here is being viewed as a property of a performance, formed through a complex and dynamic relationship between the performer, an audience and the domain in which the performance is situated [2]. It encompasses notions of expression and style within the performance alongside a demonstration of high levels of technical proficiency, a deeper understanding of the instrument, the piece being played and the context or domain of the performance.

The decision to limit the scope of this research to the domain of electric bass has been made as the author is an experienced electric bass player. There is also recent research [3, 4] within this area that can be used within this PhD.

2 Proposed Plan for Research

To address the main research question, this research will focus on developing a theory for how a computer can exhibit virtuosity within a rendered performance. To allow this theory to be tested a computer performance system that can create performances, using the physical model of electric bass guitar developed by Kramer et al. [3], is planned to be developed.

The current theory is based upon a case-based reasoning approach. Previous work on the SaxEx system [5] has demonstrated how effective case-based reasoning can be when applied to creating expressive performances. Unlike the SaxEx system, which manipulates the waveforms of a non-expressive audio recording as its output, the planned system will be manipulating physical model parameters. These parameters are intended to be abstracted to allow for rationalisation of the performances and evaluation of their virtuosity.

A performance here is being formalised in Equation 1 as the result of $Player$ applying a set of $Techniques$, $\{T_{pluck}, T_{thumb}...T_n\}$, to a sequence of $Notes$, $\langle N_1, N_2...Nn\rangle$. Musical score information, physical model and performance parameters are needed to be represented, abstracted and manipulated to produce a performance. All this information will be represented using the Common Hierarchical Abstract Representation for Music (CHARM) [7, 8].

$$Performance = Player(Techniques, Notes) \tag{1}$$

Cases are to be CHARM constituents. Constituents are formed by grouping together particles. Particles can be either events and/or other constituents. An event differs from a constituent in that it is the most fundamental element of interest within the data and as such cannot be formed from groupings. Constituents enable the formation of hierarchical structures, denoting increases in both hierarchical level and in abstraction. Events form the lowest levels of this hierarchy and within this research will be musical notes. A visual representation of an example is constituent is shown in Figure 1.

When producing a new performance, or interpreting one, a new CHARM representation will need to be constructed. First, constituents of suitable types are found, or created, and then searches for similar constituents are made. A constituent's similarity is to be judged on its structural and musical type, along with the combination and type of its particles. Retrieved constituents can the be modified by interchanging particles for better matching ones to increase the suitability of the constituent for the new case. This process of finding new constituents, then modifying them is akin to the engagement reflection cycle outlined by Pérez y Pérez [6], and is important as being able to reflect upon the performances the system creates can help to guide it towards producing virtuosic one.

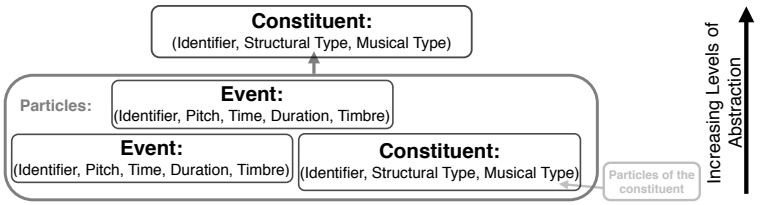


Fig. 1. A visual representation of a CHARM constituent formed of a group of three particles: two events and a constituent, (which has its own particles). I refer the reader to Smaill et al. [7] for more details on the internal structure of event and constituents.

Ontologies for domain specific knowledge e.g. musical score structure, bass technique etc. will be separate from the CHARM representation forming add-on modules for the system. To further inform the knowledge required by the system three studies are planned. One to investigate aspects of bass playing, one to collect performance data and third to see how virtuosity is perceived to inform the reflection of the system.

3 Description of Progress to date

At present I am approaching the first year review of my PhD. The work so far has been in better understanding the form the PhD will be taking, with this document forming a brief summary of the work that has been completed so far.

Acknowledgments. This work is supported by the Media and Arts Technology programme, EPSRC Doctoral Training Centre EP/G03723X/1

References

1. Widmer, G. and Goebl, W. Computational models of expressive music performance: The state of the art. JNMR, 33(3):203-216. (2004)
2. Howard, V. A.: Charm and speed: virtuosity in the performing arts. Peter Lang Publishing Inc., New York. (2008)
3. Kramer, P., Abesser, J., Dittmar, C., and Schuller, G.: A digitalwaveguide model of the electric bass guitar including different playing techniques. ICASSP, IEEE (2012)
4. Abeßer, J.: Automatic Transcription of Bass Guitar Tracks applied for Music Genre Classification and Sound Synthesis. PhD thesis, Elektrotechnik und Informationstechnik der Technischen Universität Ilmenau (2014)
5. Arcos, J. L., Mántaras, R. L., and Serra, X.: Saxex: a case-based reasoning system for generating expressive musical performances. JNMR, 27(3):194-210. (1998)
6. Pérez y Pérez, R.,. MEXICA: A Computer Model of Creativity in Writing. PhD thesis, The University Of Sussex. (1999)
7. Smaill, A., Wiggins, G., and Harris, M.: Hierarchical music representation for composition and analysis. CHum. (1993)
8. Wiggins, G., Miranda, E., Smaill, A., and Harris, M.:A framework for the evaluation of music representation systems. CMJ, 17(3):31-42. (1993)

System to design context-aware social recommender systems

Jose L. Jorro-Aragoneses

Department of Software Engineering and Artificial Intelligence
Universidad Complutense de Madrid, Spain
email: jljorro@ucm.es, belend@ucm.es, jareciog@fdi.ucm.es

Abstract. In this document, we summarize my PhD thesis goals and the progression in 2014/2015. The principal goal of my PhD thesis is to describe an architecture to design context social recommender systems. Finally, we explain all goals that we will try to achieve during my PhD studies.

1 Introduction

The number of products and the amount of information that we can consider has increased with the growth of the Internet. Sometimes, all of this information could overwhelm users. Recommender systems were created to filter this information and they just show the most interesting results for each user. For example, recommender systems are an important feature in e-commerce, where they show what products may most interest a user [8].

Recommender systems are an active research area in the artificial intelligence community. The majority of recommender systems use features of products and user preferences to calculate recommendations [7]. A trend in this area is to use contextual information [1] in recommender systems.

We find a complete definition of context in [3] *"Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves"*. In our case, entities of recommender systems are items that systems recommend and users who receive a recommendation.

My PhD thesis goal is to study what kinds of context information there are in a recommender system, how many ways we can obtain this information (implicit, users introduce the information, or explicit, the system obtains this information itself) and design a system to build recommender systems automatically.

The paper is organized as follows: Section 2 defines specific objectives in my PhD thesis based on the main goal. Finally, we explain the progress to date in Section 3.

2 Research objectives

As we said before, the main goal of my PhD thesis is to analyse what type of context information could be used in a recommender system and we will use these results to create an architecture that creates templates of CBR recommender systems automatically. To do it, we need to analyse different recommender systems and observe what type of information have its elements.

We can find 4 types of context information based on [3]:

Individual: Features of entities (age, sex, restaurant type, ...).
Location: Location of entities (longitude, latitude, room of a museum, ...).
Time: Time or time restrictions of entities (timetable, date of an event, ...).
Relationship: Features that we obtain in entities relationship (a family, a group of pictures of the same theme,...).

Currently, we are studying recommender systems that we have built before and classifying context information of items and types and type of users. Firstly, we classify *MadridLive* [2, 6], a recommender system of tourism and leisure activities in Madrid. This system uses all types of context information, and after, we add the emotional context (part of my PhD thesis) to complete the system. At the same time, we study different ways to obtain this information (mobile devices, social networks, linked-data, etc.). With context types and forms to obtain the information we create an ontology that classifies CBR recommender systems by the type of information that these systems use. Finally, we are going to use this ontology to make a system that builds templates of CBR systems. This system will use the type of items, users and technology to create a template that explains how to build the recommender system.

Preliminary specific objectives are defined as follows:

Objective 1: Detection and study of the influence of emotional context in recommender systems.
- **Objective 1.1:** Obtain a method to detect the user emotions by his/her facial gestures. *The preliminary results have been published in [5, 4].*
- **Objective 1-2:** Investigate different applications of emotions in recommender systems.

Objective 2: Classify all context information types and all different forms to obtain each type of information. To do it, we will design an ontology that will be used to create the final system. In this objective we are going to study recommender systems for individuals only.

Objective 3: Extend classification to group recommender systems. The main goal is to detect and study the social context in recommender systems.
- **Objective 3.1:** Obtain a method to calculate the influence between members of a group using social networks.
- **Objective 3.2:** Determine if there are patterns of groups with similar characteristics, for example, families, seniors group, etc.
- **Objective 3.3:** Add social context conclusions in the ontology that we have defined in **Objective 2**.

Objective 4: Design a system that uses our ontology to create templates for CBR systems. The system creates templates using the information that the recommender system will use.

Objective 5: Make experiments to validate the system and research the influence of each type of context in a recommender system. To do it, we are going to create a recommender system based in the tourism and leisure domain.

All these specific objectives permit us to study all information types that participate in a recommender system and propose a system to design recommender systems automatically.

3 Description of the progress to date

In 2014/2015, I have finished objective-1.1. I have proposed a CBR approach to infer the emotion state using images of the user's face. This method has been published in [5] and [4]. Next, we have compared the quality of our method with others, and this comparison is explained in the paper that we have published at this conference (ICBBR 2015). In this paper, we explain a possible solution to the cold-start problem. To do it, we have created specialized case bases with cases that have the same features. These features are:

- *Age*, classified in two categories, children and adults.
- *Gender*, classified in two categories, men and women.
- *Ethnic group*, classified in the ethnic group features as Japanese, European, etc.

Actually, I am studying the design of an ontology to classify recommender systems by the type of context information that they use. The objective is using the ontology in a system that creates templates of CBR systems.

References

1. Adomavicius, G., Tuzhilin, A.: Context-Aware Recommender Systems, chap. 7, pp. 217–253. Springer US (2011)
2. Agudo Calvo, B., Jorro Aragoneses, J.L., Valle Corral, J.d.: Sistema de recomendación de actividades turísticas: Madrid live (2013)
3. Dey, A.K.: Understanding and Using Context. Personal and Ubiquitous Computing 5(1), 4–7 (Feb 2001), http://dl.acm.org/citation.cfm?id=593570.593572
4. Jorro-Aragoneses, J.L., Díaz-Agudo, B., Recio-García, J.A.: Optimization of a CBR system for emotional tagging of facial expressions. In: UKCBR (2014)
5. López-de-Arenosa, P., Díaz-Agudo, B., Recio-García, J.A.: CBR tagging of emotions from facial expressions. In: ICCBR (2014)
6. López Gómez, J., Bezares Álvarez, M., Marín Fernández, R.: Madrid live: un sistema de recomendación de ocio social y contextual para la ciudad de madrid (2014)
7. Ricci, F., Rokach, L., Shapira, B.: Introduction to Recommender Systems, chap. 1, pp. 1–35. Springer (2011)
8. Schafer, J., Konstan, J., Riedl, J.: E-commerce recommendation applications. In: Kohavi, R., Provost, F. (eds.) Applications of Data Mining to Electronic Commerce, pp. 115–153. Springer US (2001)

Experience-based Recommendation for a Personalised E-learning System

Blessing Mbipom

School of Computing Science and Digital Media,
Robert Gordon University,
Aberdeen, Scotland,
United Kingdom.
b.e.mbipom@rgu.ac.uk

1 Introduction

A large amount of learning resources is available to learners on the Web. Users of these resources are often discouraged by the time spent in finding and assembling relevant resources to support their learning goals, and these users often face the information overload problem [4]. Personalisation within e-learning would allow the learning abilities and preferences of individual learners to be taken into account, thus enabling such systems to offer relevant resources to learners [7].

The interaction of previous learners with resources and the resulting outcome can be viewed as a learning experience. An experience-based recommendation approach would allow the experiences of similar users to be reused for making recommendations to new users. Currently, some recommender systems in e-commerce can capture the experience of users with items and reuse these to enhance recommendation [2]. However, little work has been done to reuse experiences in the e-learning domain [1, 3]. There is potential to improve the recommendations made within e-learning [5], drawing from the impact that the reuse of user experiences has made within e-commerce. Although recommendation in the e-learning domain is challenging given that the learning resources have to be carefully combined unlike individual products in e-commerce.

A key contribution of this research will be the development of innovative approaches to incorporate the learning experiences of previous learners captured in outcomes such as reviews and ratings, in the recommendations made to new learners. This research will harness the wide range of available e-learning resources in order to cater for learners with different preferences. The knowledge contained in the learning resources will be employed for refining learners' goals and indexing new learning resources. This work will improve the current state of e-learning systems by reusing the experiences of previous learners when recommending relevant learning resources to new learners.

2 Research Questions

This research aims to capture and reuse the learning experiences of previous learners to enhance recommendations made to new learners within a personalised e-learning system? This research seeks to address the following questions:

- How can learners' goals be refined to improve the recommendation of learning resources?
- How can learners' preferences and abilities be captured to enhance personalised recommendations?
- How can learning resources be represented to support effective retrieval?
- How can outcomes such as learners' reviews and ratings, be captured and reused to enhance e-learning recommendation?

3 Research Plan

This research will involve the development of novel approaches for reusing the experiences of previous learners to enhance e-learning recommendation. Techniques to capture learners' preferences and abilities will be developed. Existing learner models will be adapted for this task with the aim of capturing the preferences and the abilities of learners. This information would be used for making relevant recommendations to new learners.

Existing knowledge sources will be organised into a coherent background knowledge structure. Potential knowledge sources such as Microsoft Academic Search, the ACM Computing Classification System, and Wikipedia have already been identified. The plan is to employ these in the development of a background knowledge structure which can be employed for refining learners' goals and for indexing learning resources. This structure will be useful for identifying the links between resources and for recommending relevant resources.

Methods for representing and refining learners' goals will be developed. This is necessary in e-learning because learners often have insufficient knowledge of the domain to formulate suitable goals. The plan is to map the goals to a resource representation developed using shared background knowledge, this will entail reasoning with the text in the goals and the learning resources.

Representations that capture learners' outcomes will be created. Learners' test scores, reviews and ratings can be viewed as outcomes in an e-learning domain. Currently, learners' test scores are the major form of feedback used in e-learning. However, this does not capture learners' opinions which can be effectively employed to inform other learners. The plan is to incorporate user-opinions with user-performance to enhance the recommendation process.

4 Current Progress

The research methodology has been substantially developed. Various approaches for representing learning resources have been investigated, these range from

knowledge-light to knowledge-rich approaches. Some methods of refining learners' goals have also been examined.

Different types of learning resources have been identified to use as data for this work. These include e-books, online teaching slides and video lectures. They have been chosen because they contain structure and metadata that will help with the research, and because of the variety of media types contained.

Preliminary experiments have been carried out to develop a background knowledge structure to use for the refinement of learners' goals and the representation of new learning resources. A collection of 217 e-book chapters from the machine learning domain were collected for the experiments. Terms and phrases were extracted from the Tables-of-contents (TOCs) of the e-books using some NLP techniques and phrase identification methods.

E-books are used as the primary data source in this work because of the structure they contain and because they are designed to be effective for teaching and learning. Furthermore, issues of trust and provenance [6] are catered for because the nature of books means an author and affiliation exists. Wikipedia is used as a complementary data source, because it is a knowledge-rich source put together by many contributors.

Terms and phrases were extracted from the TOCs of e-books and compared with phrases from the Machine Learning category in Wikipedia to generate a set of suitable phrases to use for developing the background knowledge structure. The result was 90 phrases consisting of 17 unigrams, 58 bigrams and 15 trigrams.

Initial output shows the potential to harness the knowledge in e-Books and Wikipedia for developing a background knowledge structure that will enable the refinement of learners' goals and indexing of new learning resources. Further work will involve evaluation of this method, and the development of a system that employs the background structure to recommend relevant learning resources.

References

1. Bobadilla, J., Hernando, A., Arroyo, A.: E-learning experience using recommender systems. In: Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, pp. 477–482. ACM (2011)
2. Dong, R., Schaal, M., O'Mahony, M.P., McCarthy, K., Smyth, B.: Opinionated product recommendation. In: S.J. Delany, S. Ontañón (eds.) Case-Based Reasoning Research and Development, *LNCS*, vol. 7969, pp. 44–58. Springer, Heidelberg (2013)
3. Ghauth, K.I.B., Abdullah, N.A.: Building an e-learning recommender system using vector space model and good learners average rating. In: Ninth IEEE International Conference on Advanced Learning Technologies, pp. 194–196. IEEE (2009)
4. Kantor, P.B., Rokach, L., Ricci, F., Shapira, B.: Recommender systems handbook. Springer (2011)
5. Kolodner, J.L., Cox, M.T., González-Calero, P.A.: Case-based reasoning-inspired approaches to education. The Knowledge Engineering Review **20**(3), 299–303 (2005)
6. Leake, D., Whitehead, M.: Case provenance: The value of remembering case sources. In: Case-Based Reasoning Research and Development, pp. 194–208. Springer (2007)
7. Peter, S.E.: The use of tagging to support the authoring of personalisable learning content. Ph.D. thesis, University of Greenwich (2012)

Workflow Adaptation in Process-oriented Case-based Reasoning

Gilbert Müller

Business Information Systems II
University of Trier
54286 Trier, Germany
muellerg@uni-trier.de,
http://www.wi2.uni-trier.de

Workflows are an important research domain, as they are used in many application areas, e.g., there are business workflows, scientific workflows, workflows representing information gathering processes, or cooking instructions. Workflows are "the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules" [4]. Thus, workflows consists of a structured set of tasks and data objects shared between those tasks. In this regard, Process-oriented Case-based Reasoning (POCBR) [7] addresses the creation and adaptation of processes that are, e.g., represented as workflows. Although, POCBR is of high relevance little research exist so far.

The presented research focuses on the development of new workflow adaptation approaches and related topics, for instance the retrieval of workflows. Methods are investigated, which automatically learn adaptation knowledge from the case base. This prevents limited adaptation capabilities due to the acquisition bottleneck for adaptation knowledge.

1 Research Questions

This section presents the research questions addressed by my doctoral thesis in note form.

1. How can workflows be efficiently retrieved?
2. How can workflows be adapted regarding defined preferences or restrictions?
3. How can interactive workflow adaptation be realized?
4. How can the adaptability of workflows be reflected during retrieval?
5. How can adaptation knowledge be revised to address the retainment of adaptation knowledge?

The approaches to address the first two research questions are described in the next section and section 3 describes how the remaining open research questions are going to be investigated.

2 Current state of research

The presented research is implemented and evaluated using the CAKE (Collaborative Agent-based Knowledge Engine) framework[1] developed at the University of Trier. It deals with semantic workflows and is able to compute the similarity between two workflows according to the semantic similarity [2]. The approaches will are illustrated and investigated in the cooking domain, i.e., the workflows represent cooking recipes.

Currently, approaches addressing the first two research questions have been investigated:

1. Based on research about clustering of workflows [3], the problem of improving retrieval performance by developing a cluster-based retrieval method for workflows [8] was addressed. To achieve this, a new clustering algorithm, which constructs a binary tree of clusters was developed. The binary tree is used as index structure during a heuristic search to identify the most similar clusters containing the most similar workflows in a top-down fashion. Further, POQL [12] was developed serving as query language to guide the retrieval and the adaptation of workflows regarding defined preferences or restrictions.
2. Several adaptation approaches had been investigated to address the second research question. A compositional adaptation approach for workflows was investigated [9] where workflows are decomposed into meaningful subcomponents, called *workflow streams*. In order to support adaptation, streams of the retrieved workflow are replaced by appropriate streams of other workflows. Based on this work, operator-based adaptation [11] has been developed. The adaptation operators are learned automatically based on the workflows in the case base enabling to remove, insert or replace workflow fragments. Further, workflow generalization and specialization has been addressed [10], which increases the coverage of the workflow cases and thus being able to support adaptation as well.

3 Future Work

In future work, an additional adaptation approach will be investigated for semantic workflows, similar to the adaptation approach presented by Minor et. al. [6], which is based on adaptation cases describing how to transform a particular workflow to a target workflow. The future work addressing the remaining research questions 3.-5. is summarized below.

A drawback of applying traditional adaptation methods is that the adaptation goal must mostly be known previously. Consequently, this can lead to a non-optimal or not desired solution. Hence, interactive adaption [1] will be investigated, as it is a promising approach to overcome this drawback. It supports

[1] cakeflow.wi2.uni-trier.de

the search of a suitable query and hence the desired solutions by involving user interaction during adaptation.

Further, separating similarity-based retrieval and adaptation may provide workflows that can not be at best adapted according to the query. Hence, methods will be developed that also reflect the adaptability of the workflows during the retrieval stage [13].

Moreover, feedback of workflow adaptation will be captured in order to address the retaining of adaptation knowledge [5]. This is essential, as the quality of automatically learned adaptation knowledge can not always be ensured. Thus, the quality of workflow adaptation is improved. Further, the growth of adaptation knowledge can be controlled and hence the performance of adaptation can be maintained.

References

1. Aha, D.W., Muñoz-Avila, H.: Introduction: Interactive case-based reasoning. Applied Intelligence 14(1), 7–8 (2001)
2. Bergmann, R., Gil, Y.: Similarity assessment and efficient retrieval of semantic workflows. Inf. Syst. 40, 115–127 (Mar 2014)
3. Bergmann, R., Müller, G., Wittkowsky, D.: Workflow clustering using semantic similarity measures. In: Timm, Thimm (eds.) KI 2013: Advances in Artificial Intelligence. LNCS, vol. 8077, pp. 13–24. Springer (2013)
4. Hollingsworth, D.: Workflow management coalition glossary & terminology. `http://www.wfmc.org/docs/TC-1011_term_glossary_v3.pdf` (1999), last access on 04-04-2014
5. Jalali, V., Leake, D.: On retention of adaptation rules. In: Case-Based Reasoning Research and Development, pp. 200–214. Springer (2014)
6. Minor, M., Bergmann, R., Görg, S., Walter, K.: Towards case-based adaptation of workflows. In: Case-Based Reasoning. Research and Development, pp. 421–435. Springer (2010)
7. Minor, M., Montani, S., Recio-Garca, J.A.: Process-oriented case-based reasoning. Information Systems 40(0), 103 – 105 (2014)
8. Müller, G., Bergmann, R.: A Cluster-Based Approach to Improve Similarity-Based Retrieval for Process-Oriented Case-Based Reasoning. In: Proceedings of ECAI 2014. Prague, Czech Republic (2014)
9. Müller, G., Bergmann, R.: Workflow Streams: A Means for Compositional Adaptation in Process-Oriented CBR. In: Proceedings of ICCBR 2014. Cork, Ireland (2014)
10. Müller, G., Bergmann, R.: Generalization of Workflows in Process-Oriented Case-Based Reasoning. In: 28th International FLAIRS Conference. AAAI, Hollywood (Florida), USA (2015)
11. Müller, G., Bergmann, R.: Learning and Applying Adaptation Operators in Process-Oriented Case-Based Reasoning. In: Proceedings of ICCBR 2015. Frankfurt, Germany (2015)
12. Müller, G., Bergmann, R.: POQL: A new query language for Process-Oriented Case-Based Reasoning (2015), to be submitted to LWA 2015 Trier, Germany
13. Smyth, B., Keane, M.: Retrieving adaptable cases. In: Wess, S., Althoff, K.D., Richter, M. (eds.) Topics in Case-Based Reasoning, Lecture Notes in Computer Science, vol. 837, pp. 209–220. Springer Berlin Heidelberg (1994)

Opinionated Explanations of Recommendations from Product Reviews

Khalil Muhammad

Insight Centre for Data Analytics,
University College Dublin, Belfield, Dublin 4, Ireland.
{firstname.lastname}@insight-centre.org

1 Introduction

Recommender systems are now mainstream and people are increasingly relying on them to make decisions in situations where there are too many options to choose from. Yet many recommender systems act like "black boxes", providing little or no transparency into the rationale of their recommendation process [1]. Related research in the field of recommender systems has focused on developing and evaluating new algorithms that provide more accurate recommendations. However, the most accurate recommender systems may not necessarily be those that provide the most useful recommendations — due to the influence of how recommendations are presented and justified to users [2–4]. Therefore, recommender systems must be able to explain what they do and justify their actions in terms that are understandable to the user. An explanation, in this context, is any added information presented with recommendations to help users better understand why and how a recommendation is made [5]. Studies show that explanations help users make better decisions and are therefore provided for many reasons [6, 7], which normally align with the objective of the recommender system. Interestingly, explanations may sometimes be provided from the users (not from the recommender system) to justify their choices [8].

The availability of user-generated reviews that contain real experiences provides a new opportunity for recommender systems; yet, existing methods for explaining recommendations hardly take into account the implicit opinions that people express in such reviews even though studies show that users are increasingly relying on the reviews to make better choices [9]. Also, explanations usually provide a posthoc rationalisation for recommendations; but, this work is motivated by a more intimate connection between recommendations and explanations, which poses the question: *can the recommendation process itself be guided by structures generated to explain recommendations to users?*

This work builds on existing research in the areas of case-based reasoning, recommender systems and opinion mining to propose a novel approach for building explanations in recommender systems. We will also explore the potential of opinionated explanations in driving the recommendation process.

2 Research Plan

The core focus of this work is to explore the role of opinions in explaining recommendations. Accordingly, we have identified the following areas of interest:

Ranking, filtering and evaluating feature quality. Feature-level opinion mining algorithms that are capable of extracting very granular opinions, such as [10], yield noisy features because they rely on shallow natural language processing (NLP) techniques. Ultimately, these features lack context and are too fine-grained to be intuitive to users. For instance, it will be nonsensical to explain a hotel recommendation as *"because visitors liked the wire..."*, where *'wire'* is a feature mined from reviews. Hence, the research question: *"how to rank, filter and evaluate features mined from reviews"*. We will use off-the-shelf opinion mining techniques but focus on developing methods for ranking features so that they can be filtered and evaluated for quality (i.e. the extent to which a feature is relevant and presentable to users in explanations). This involves creating new methods for summarising features so that only qualitative and comprehensible features are presented in explanations.

Generating opinionated explanations. Explanations normally demonstrate how one or more recommended items relate to a user's preferences, normally through an intermediary entity such as a user, item, or feature. For instance, Netflix may use the movies that a user has rated highly in the past to explain a movie recommendation. And since user ratings are often unable to fully represent user preferences, there is a place of fine-grained opinions that are explicitly provided by users in textual reviews. We expect that explanations that are based on opinionated reviews will be more natural and convincing. Hence the research question: *how to use such opinions to generate explanations of product recommendations?*. We will use opinions from reviews to generate that justify a particular recommendation or sets of recommendations, and we will conduct live-user trials to test for its usefulness in decision-making.

Driving recommendations using explanations. To date, most recommender systems have treated explanations as an afterthought, presenting them alongside recommendations, but with little connection to the recommendation process itself. This work will explore the potential of using explanations to drive the recommendation process itself so that, for example, an item will be recommended because it can be explained in a compelling way. Hence the research question: *how to use explanations to support similarity metrics and ranking strategies in a recommendation process?*

3 Progress

To address the problem of feature quality, we used the approach in [10] to mine opinions from a dataset of TripAdvisor hotel reviews. Then, using various lexical and frequency-based filtering techniques, we removed noisy, less opinionated and unpopular features. The remaining features were summarised into higher-level representations by clustering them based on the words they co-occur with in

sentences of reviews. This feature representation allows us to replace a low-level feature (e.g. 'orange juice') with a more meaningful higher-level one (e.g. 'breakfast') that is suitable for use in explanations.

We developed a new method for generating personalized explanations which highlight the pros and cons of a recommended item to a user. Our approach focuses on the features that the user has mentioned in their reviews, and those mentioned about the recommended item by other users. In the explanation, we prioritize the features that are likely to be of interest to the user. Each feature is classified as a pro or con based on its sentiment, and it is ranked by its popularity with the user and the recommended item.

We also developed another explanation strategy that explains a recommended item in comparison with other recommendations. That is, the explanation presents features of the recommended item that are better or worse than its alternatives.

Acknowledgments. This work is supported by the Insight Centre for Data Analytics under grant number SFI/12/RC/2289.

References

1. Herlocker, J.L., Konstan, J.A., Riedl, J.: Explaining Collaborative Filtering Recommendations. In: Proceedings of the 2000 ACM Conference on Computer supported cooperative work, ACM (2000) 241–250
2. McNee, S.M., Riedl, J., Konstan, J.A.: Being Accurate is not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In: CHI'06 extended abstracts on Human factors in computing systems, ACM (2006) 1097–1101
3. Pu, P., Chen, L., Hu, R.: Evaluating recommender systems from the users perspective: survey of the state of the art. User Modelling and User-Adapted Interaction **22**(4-5) (2012) 317–355
4. Knijnenburg, B.P., Schmidt-Thieme, L., Bollen, D.G.: Workshop on User-centric Evaluation of Recommender Systems and their Interfaces. In: Proceedings of the fourth ACM conference on Recommender systems, ACM (2010) 383–384
5. Tintarev, N., Masthoff, J.: Effective Explanations of Recommendations: User-Centered Design. In: Proceedings of the 2007 ACM conference on Recommender systems, ACM (2007) 153–156
6. Tintarev, N., Masthoff, J.: Designing and Evaluating Explanations for Recommender Systems. In: Recommender Systems Handbook. Springer (2011) 479–510
7. Zanker, M.: The Influence of Knowledgeable Explanations on Users' Perception of a Recommender System. In: Proceedings of the sixth ACM conference on Recommender systems, ACM (2012) 269–272
8. Nunes, I., Miles, S., Luck, M., De Lucena, C.J.: Investigating Explanations to Justify Choice. In: User Modeling, Adaptation, and Personalization. Springer (2012) 212–224
9. Lee, J., Park, D.H., Han, I.: The Different Effects of Online Consumer Reviews on Consumers' Purchase Intentions Depending on Trust in Online Shopping Malls: An Advertising Perspective. Internet research **21**(2) (2011) 187–206
10. Dong, R., Schaal, M., O'Mahony, M.P., McCarthy, K., Smyth, B.: Opinionated Product Recommendation. In: Case-Based Reasoning Research and Development. Volume 7969. (2013) 44–58

Integrated Maintenance with Case Factories for distributed Case-Based Reasoning Systems

Pascal Reuss[1,2]

[1] German Research Center for Artificial Intelligence
Kaiserslautern, Germany
http://www.dfki.de
[2] Institute of Computer Science, Intelligent Information Systems Lab
University of Hildesheim, Germany
http://www.uni-hildesheim.de

1 Integrated maintenance with Case Factories

Maintenance approaches for Case-Based Reasoning (CBR) systems focus on a single CBR system and mainly on a single knowledge container like the case base, the adaptation knowledge or the similarity. There are a few approaches that deal with shifting knowledge between knowledge containers, especially between the case base and adaptation knowledge. These approaches have been successfully applied to CBR systems in the past. In many systems, especially multi-agent systems, the knowledge is distributed among several homogenous or heterogeneous knowledge sources. Therefore, a maintenance approach has to consider the dependencies between these knowledge sources as well as high-level maintenance goals. An example is removing one or more cases from a case base. Cases in other CBR systems could depend on one of the removed cases, so they may become inconsistent (to some degree). The system should suggest an appropriate maintenance action like removing the depending cases to keep the system's correctness/consistency. To address these dependencies between CBR systems and their knowledge containers, a new maintenance approach for distributed CBR systems is required. In the following, I will describe the idea of an integrated maintenance approach based on so-called Case Factories, that will be capable of generating a maintenance plan for multiple CBR systems, considering dependencies, and providing explanations for the generated maintenance actions. The approach is designed for multi-agent systems based on the SEASALT architecture ([2]), which supports distributed knowledge sources. To develop this approach, four research goals have to be reached:

- Extend the original Case Factory approach from single system support to multi system support.
- Integrate maintenance explanation capabilities into the new approach
- Develop a methodology to apply the new approach to existing multi-agent systems as well as integrating it into the development of new multi-agent systems
- Evaluate the new approach and the methodology within an industrial use case for a decision support system

1.1 Case Factory and Case Factory Organization

The original idea of the Case Factory (CF) is from Althoff, Hanft and Schaaf ([1]) and is based on the Experience Factory approach from software engineering. A CF consists of several software agents for different tasks like evaluating incoherence or modifying the case base. Each knowledge source, in this context CBR systems, has its own CF that is responsible for maintaining the dedicated knowledge. The original approach has to be extended to support the maintenance of the other three knowledge containers, namely vocabulary, similarity and adaptation knowledge, considering intra-system dependencies between the knowledge containers. The original approach contains several software agents to monitor the case-base and one agent to do the necessary maintenance actions. To support all knowledge containers reasonably some more agents are required to monitor these containers and the maintenance tasks should be split on several agents. An own maintenance agent per knowledge container is required to support parallel maintenance of the knowledge. Additionally, a supervising agent is required to coordinate the maintenance actions. This agent is also responsible for the communication between the multiple CFs. Monitoring a knowledge container means to notify the supervising agent about changes of the knowledge inside the container, e.g. removing a case, renaming a concept, or adding a rule. Evaluation of a knowledge container means to check a knowledge container for inconsistencies, problem solving competence or efficiency. Therefore, different evaluation strategies could be used, for example computing the coverage and reachability of cases ([9]) or inconsistency checking ([8], [7] and [6]).

To store information about dependencies, maintenance goals and evaluation criteria, a so-called Maintenance Map is used. The Maintenance Map is based on the Knowledge Map from Davenport and Prusak ([4]), which was adapted to multi-agent-systems by Bach et al. ([3]). In contrast to a Knowledge Map, the Maintenance Map is a bidirectional graph. The vertices represent the knowledge sources in a distributed knowledge-based system and the edges represent the dependencies between the single sources. The weights of the edges could be used to describe the importance of the dependency. Additional information like maintenance goals, metrics, thresholds, or constraints could be defined, too. The Maintenance Map can also contain information about the priority of maintenance actions or about associated maintenance actions. This may be helpful, when a combination of maintenance actions is necessary to preserve the competence or efficiency of a single CBR system or the MAS ([5]).

While a CF is able to maintain a single CBR system, a high-level Case Factory Organization (CFO) is required to coordinate the actions of all CFs and take the dependencies between the single CBR systems into account. This CFO consists of several additional software agents to supervise the communication between the CFs and the adherence of high level maintenance goals. Additionally, agents collect the maintenance suggestions from the CFs and derive a maintenance plan from all single maintenance suggestions. The agents are also responsible for checking constraints or solving conflicts between individual maintenance suggestions. In addition, a maintenance suggestion may trigger follow-up maintenance actions based on the dependencies between the CBR systems. The concept of the CFO allows to realize as many CFs and layers of CFOs as required. This way a hierarchy of CFOs can be established that is scalable and supports multi-agent systems with many agents and layers.

To support the knowledge engineer, the CF and CFO should explain, why a certain maintenance action was suggested. To give a CBR system explanation capabilities a lot of knowledge is necessary. The underlying research assumption here is that the minimal knowledge required for the explanation of maintenance actions is the same knowledge that is required to suggest a maintenance action. It follows, that the minimal knowledge for explanations already exists in the system, if the system is able to (reasonably) suggest maintenance actions.

1.2 Current state of research and research plan

Currently, the concepts of the CF and the CFO are defined. An initial version of a CF is implemented within a research system called docQuery in the travel medicine domain. This implementation has to be extended with a CFO to test the integrated maintenance approach. The next steps on conceptual level are to formalize the concept and advance the maintenance planning and the explanation strategies on a more detailed level and implement them, too. It is planned to complete the conceptual development of the integrated maintenance approach this year and implement the complete approach within the docQuery system. Next year, the approach will be applied to an industrial use case in the domain of aircraft diagnosis to test the approach within an industrial environment. The evaluation of the approach and methodology will be completed till the third quarter of 2016 and the dissertation will be written until the beginning of 2017.

References

1. Althoff, K.D., Hanft, A., Schaaf, M.: Case factory - maintaining experience to learn. Advances in Case-Based Reasoning Lecture Notes in Computer Science 4106/2006, 429–442 (2006)
2. Bach, K.: Knowledge Acquisition for Case-Based Reasoning Systems. Ph.D. thesis, University of Hildesheim (2013), dr. Hut Verlag Mnchen
3. Bach, K., Reichle, M., Reichle-Schmehl, A., Althoff, K.D.: Implementing a coordination agent for modularised case bases. In: Proceedings of the 13th UK Workschop on Case-Based Reasoning. pp. 1–12 (2008)
4. Davenport, T.H., Prusak, L.: Working Knowledge: How Organizations Manage What they Know. Havard Business School Press (2000)
5. Leake, D., Kinley, A., Wilson, D.: Learning to integrate multiple knowledge sources for case-based reasoning. In: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence. pp. 246–251. Morgan Kaufmann (1997)
6. McSherry, D.: An adaptation heuristic for case-based estimation. In: 4th European Workshop, EWCBR-98. pp. 184–195 (1998)
7. Racine, K., Yang, Q.: Maintaining unstructured case bases. In: Second International Conference on Case-Based Reasoning. pp. 553–564 (1997)
8. Smyth, B.: Case-base maintenance. In: 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems IEA-98-AIE. pp. 507–516 (1998)
9. Smyth, B., Keane, M.: Remembering to forget: A competence-preserving case deletion policy for case-based reasoning systems. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence. pp. 377–382 (1995)