LSIR-2

Logic and the Simulation of Interaction and Reasoning

Pasadena CA, U.S.A., 12 July 2009
edited by Benedikt Löwe

Logic and the Simulation of Interaction and Reasoning: Logic, Computer Games, and Narrative

Benedikt Löwe
Institute for Logic, Language and Computation, Universiteit van Amsterdam,
Postbus 94242, 1090 GE Amsterdam, The Netherlands
Department Mathematik, Universität Hamburg, Bundesstrasse 55, 20146 Hamburg, Germany
Mathematisches Institut, Rheinische Friedrich-Wilhelms-Universität Bonn,
Endenicher Allee 60, 53115 Bonn, Germany

Logic & Interaction

In recent years, logicians have become more and more interested in the phenomenon of interaction. The research area *Logic and Games* deals with the transition from the static logical paradigm of formal proof and derivation to the dynamic world of intelligent interaction and its logical models. Fruitful technical advances have led to deep insights into the nature of communicative interaction by logicians. This new direction of logic has quickly gained momentum and support. In 2006, a Marie Curie Research Training Site GLoRiClass ("Games in Logic Reaching Out for Classical Game Theory") was opened in Amsterdam, providing graduate student training for a number of PhD students. In 2007, the European Science Foundation (ESF) has recognized this direction as one of the foremost research developments for the European science community and created a collaborative research platform called "LogICCC — Modelling intelligent interaction". Finally, in December 2007, a new book series entitled "Texts in Logic and Games" was launched.

While these interactive aspects are relatively new to logicians, on a rather different level, modelling intelligent interaction has been an aspect of the practical work of computer game designers for a long time. Pragmatic questions such as 'What makes a storyline interesting', 'What makes an reaction natural', and 'What role do emotions play in game decisions' have been tackled by practicing programmers. The practical aspects of computer gaming reach out to a wide interdisciplinary field including psychology and cognitive science. So far, there are only a few cross-links between these two communities.

LSIR-1 in Aberdeen (April 2008)

When the need for a venue to connect the research in logic and games with the more applied research on stories, games and simulation of behaviour was recognized, the mentioned research training site GLoRiClass decided to organize a workshop *Logic and the Simulation of Interaction and Reasoning* at the AISB Convention in Aberdeen in April 2008 (for more on the motivation for this workshop, cf. (Löwe 2008)). The programme committee for this workshop consisted of Stefania Bandini, Johan van Benthem, Cristiano Castelfranchi, Bruce Edmonds, Jaap van den Herik, Wiebe van der Hoek, Benedikt Löwe, Yoav Shoham, Keith Stenning, and Rineke Verbrugge. A number of research connections opened up during the workshop and discussions between researchers in the gaming industry and logicians created the opportunity for new ideas: LSIR-1 was considered a success by the participants (the proceedings of the workshop are available as (Guerin, Löwe, and Vasconcelos 2008)). The general sentiment of participants was that such a workshop should be repeated, possibly with more focus, enabling directed efforts for new research projects.

The Goals of LSIR-2

The successor of the Aberdeen workshop is now being held at IJCAI 2009, this time concretely focussing on the relation between modern techniques in logic (such as discourse representation theory or dynamic epistemic logic) and concrete modelling problems in computer games (either as part of the story or game design or as part of the design of the artificial agents).

As with LSIR-1, our workshop is sponsored by the Research Training Site GLoRiClass, allowing us to invite three speakers (Morgenstern, Schubert, and Young). Talks at LSIR-2 will describe either modelling techniques with potential applications to questions in computer game design or story design or to describe modelling problems that might be an area of application for techniques from logic. The workshop layout allows ample time for informal discussions. The experience from Aberdeen shows that this time is used properly and efficiently and will result in opening up research opportunities between the fields.

Programme Committee.

The programme committee of LSIR-2 consisted of Jan Broersen (Utrecht, The Netherlands), Cristiano Castelfranchi (Rome, Italy), Frank Dignum (Utrecht, The Netherlands), Anton Leuski (Marina del Rey CA, United States of America), Benedikt Löwe (Amsterdam, The Netherlands), Erik T. Mueller (Hawthorne NY, United States of America), Amitabha Mukherjee (Kanpur, India), Mark Overmars (Utrecht, The Netherlands), Eric Pacuit (Stanford CA,

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 3–4.

United States of America), Rohit Parikh (New York NY, United States of America), David Traum (Marina del Rey CA, United States of America), Jos Uiterwijk (Maastricht, The Netherlands), and Hans van Ditmarsch (Otago, New Zealand & Aberdeen, Scotland). We would like to thank the members of the PC for their effort and input.

Schedule

LSIR-2 will have ten presentations: three 45-minute invited presentations and seven contributed talks of 20 minutes. There were two more accepted presentations that had to be cancelled due to travel restrictions of the presenters. These were a paper entitled "Strategies made explicit in Dynamic Game Logic" by Sujata Ghosh (Groningen) and a paper by Jan Broersen (Utrecht). The latter paper is represented in this abstract volume.

09.05–09.15 Opening

09.15–10.00 R. Michael Young. *The Representational Challenges of Fictional Worlds.*

10.00–10.30 *Coffee Break*

10.30–10.50 Amitabha Mukerjee. *Discovering symbols from interactions — easier than explaining interactions via symbols?*

10.50–11.10 Nadine Guiraud, Andreas Herzig and Emiliano Lorini. *Speech acts as announcements.*

11.10–11.30 *Break*

11.30–12.15 Lenhart Schubert. *From generic sentences to scripts.*

12.15–14.20 *Lunch Break*

14.40–15.00 Jos Uiterwijk and Kevin Moesker. *Mathematical Modelling in TwixT.*

15.00–15.30 *Coffee Break*

15.30–15.50 Rafael Pérez y Pérez. *Emotions in Plot Generation.*

15.50–16.10 Ethan Kennerly. *Computing Quality of Life in a Social Management Game*

16.10–16.30 Ethan Kennerly, Andreas Witzel and Jonathan Zvesper. *Thief Belief*

16.30–16.50 Break

16.50–17.10 Martin Magnusson, David Landén and Patrick Doherty. *Logical Agents that Plan, Execute, and Monitor Communication*

17.10–17.55 Leora Morgenstern. *Traveling Through Time and Logical AI: Toward a Formal Theory of Time Travel*

References

Guerin, F.; Löwe, B.; and Vasconcelos, W., eds. 2008. *Proceedings of the AISB 2008 Symposium on Logic and the Simulation of Interaction and Reasoning*, volume 9 of *AISB 2008 Convention, Communication, Interaction and Social Intelligence, 1st-4th April 2008, University of Aberdeen.*

Löwe, B. 2008. Logic and the simulation of interaction and reasoning: Introductory remarks. In Guerin, F.; Löwe, B.; and Vasconcelos, W., eds., *Proceedings of the AISB 2008 Symposium on Logic and the Simulation of Interaction and Reasoning*, volume 9 of *AISB 2008 Convention, Communication, Interaction and Social Intelligence, 1st-4th April 2008, University of Aberdeen*, iii–v.

The Representational Challenges of Fictional Worlds

R. Michael Young
Department of Computer Science, North Carolina State University,
Raleigh, NC 27695-7534, United States of America

Introduction and Background

Modern computer games often have a strong narrative structure to the progress of their gameplay. In some genres, story is at the forefront of the interaction. As game makers seek to add new capabilities for procedurally generated content to their new releases, work on interactive narrative is growing in significance both as a basic research topic as well as a source for insight into the creation of content for games. As a result, there are now many compelling challenges for researchers working in AI that involve interactive computer games as their area of application. A significant different exists between application contexts familiar to most AI researchers and those found in games, however. Games are *fictional worlds*. Of course, we are all aware that games are fictional worlds. What is not so obvious is what this means to the relationship between a player and his or her game system and what consequences the relationship holds for the use of representations like logic and other formal computational approaches. Games are expected to be fictional in much the same way that other narrative media are. Their worlds share many properties in common with the real world, but their experience is specifically not realistic. Games are not simulations, where system behaviors are required to be identical to real world correlates. When a game player begins to play a game, he or she steps inside a magic circle (Huizinga 1955), effectively entering into a contract with the game? designers where players agree to suspend disbelief and play according to the rules and designers agree to manipulate the game world in ways that facilitate that effective play rather than recreate realistic interaction.

Understanding narrative in these game worlds, as in other media, involves the active construction and experience of an illusion (Gerrig 1993; Johnston and Thomas 1995). This construction process is particularly participatory in computer games, where players not only work to comprehend the story but take action throughout its unfolding to shape it in significant ways.

The distinction that I am drawing between simulations and game worlds points to interesting challenges for AI researchers in the representation and modeling of players?interactions with games. I describe two examples below, but essentially the larger class of challenges revolves around the requirement not to create a real world experience in a game, but to create a fictional world whose experience will prompt the same narrative comprehension processes that people use to understand other forms of narrative in conventional media (Graesser, Mills, and Zwaan 1997).

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 5–7.

Levels of Narrative

When considering the structure of narrative, narrative theorists typical break a narrative down into two distinct levels: the story and the discourse (Chatman 1980). The story level contains the things within the story world itself ?the characters and their mental state, the characters actions, the objects and locations they interact with and within. The discourse contains those medium-specific elements used by the story teller to communication the story to the reader ?a novel? text, a film? camera shots and shot sequences, background music and lighting. This division is useful for approaching the design of narrative generation systems as it provides a direct and reasonable way of factoring the many problems (Young 2007). The discussion below is divided into two parts, providing an example representational challenge from each level.

Two Representative Challenges

At the Story Level: Conflict

Conflict (or rather, the absence of it) is one of the most significant representational challenges to current work on interactive narrative. This is particularly true of plan-based approaches to narrative generation (e.g., (Cavazza, Charles, and Mead 2002; Young and Riedl 2005)), which rely on provably sound planning systems to create flaw-free (and thus conflict-free) character plans. In contrast to the plans produced by AI planning systems, stories are delightfully full of conflict. Logical systems hoping to model stories must explicitly deal with the representation of this conflict as a fundamental element of story structure. Like most real-world notions that are found within stories, narrative conflict is both like and unlike its real world correlate. As in the real world, conflict in narrative typically involves two or more characters each holding goals that are in opposition to one another who form plans both to achieve their own goal and often to prevent their opponents from achieving theirs. Narrative conflict differs from real-world conflict, however, in a number of ways. For instance, narrative conflict is paced, so that conflict is interleaved in the story between periods of calm where other elements of the story can be advanced. Narrative conflict is balanced, so that protagonist and antagonist are relatively equally matched in strength. As a result, interaction between the two doesn? result in one side overwhelming the other, thus ending the story prematurely. Narrative conflict is measured in order to create a story? rising action, that is, the tension that builds as the lead in to the story? major climax.

At the Discourse Level: The Dynamics of User Beliefs

As in other forms of work focusing on the generation of discourse (Moore and Paris 1993; Maybury 1992; Hovy 1989), the generation of narrative discourse has as its goal the creation of a sequence of communication actions that effectively manipulate the beliefs of a reader (or, in our case, a player). In typical existing AI approaches to discourse generation (often focusing on instructional text as a target genre), the content and organization of a discourse is selected in order to provide rhetorical structure that leads a reader to come to hold a certain set of beliefs. These beliefs are the central communicative goals of the discourse and, in some sense, the satisfaction of these goals is all that matters in the process of constructing the discourse.

In contrast, the beliefs that a reader or player holds about the story world at the end of a narrative are much less significant than in the case of instructional text generation. Rather, a storyteller (or a storytelling system) must give significant care to the creation of a discourse that manipulates the beliefs of the reader about the story world throughout the course of the unfolding story. In narrative discourse, information is intentionally withheld in order to create effects like surprise and suspense (Ohler and Neiding 1996; Cheong and Young 2008). Information is presented out of order (e.g., flashbacks or foreshadowing) in order to drive expectations and build anticipation. Structures in narrative discourse follow specific, genre- and medium-specific conventions or idioms (Jhala and Young 2006) that license certain types of inferences in much the same way that indirect speech acts license inferences in conversational discourse. A narrative discourse may intentionally prompt inferences at one point in the story that are in direct contradiction with prompted inferences at other points in the story. In many ways, it is the trajectory of the beliefs of a story? reader that matters in narrative rather than his or her beliefs about the story at its end.

Discussion

In the discussion above, I briefly describe two representational challenges for interactive narrative within game worlds. Both these challenges arise from the particular fictional nature of game worlds themselves and the important difference between real-world and narrative models. These two areas are provide a significant number of open research question, but they are not the only areas where progress on fictional representation is needed. Fortunately, there are many familiar logical tools that can be brought to bear on these issues. Epistemic logics, dynamic logics, logics of action and change and other computational models with well-understood semantics address aspects of these problems. Just as planning algorithms provide significant, well-founded models for the generation of casual and temporal structures in stories, so these logical tools come with obvious first steps towards application in narrative contexts. Whatever points are chosen for advance in this area, however, the significantly distinct nature of narrative and fictional worlds must play a central and consistent role.

References

Cavazza, M.; Charles, F.; and Mead, S. J. 2002. Planning characters' behaviour in interactive storytelling. *The Journal of Visualization and Computer Animation* 13:121–131.

Chatman, S. 1980. *Story and Discourse: Narrative Structure in Fiction and Film*. Cornell University Press.

Cheong, Y., and Young, R. M. 2008. Narrative generation for suspense: Modeling and evaluation. In *Proceedings of the International Conference on Interactive Digital Storytelling*.

Gerrig, R. 1993. *Experiencing Narrative Worlds: On the Psychological Activities of Reading*. Yale University Press.

Graesser, A.; Mills, K. K.; and Zwaan, R. A. 1997. Discourse comprehension. *Annual Review of Psychology* 48.

Hovy, E. 1989. Approaches to the planning of coherent text. In Paris, C.; Swartout, B.; and Mann, W., eds., *Natural Language Generation in Artificial Intelligence and Computational Linguistics*.

Huizinga, J. 1955. *Homo Ludens: A Study of the Play-Element in Culture*. Beacon Press.

Jhala, A., and Young, R. M. 2006. Representational requirements for a plan based approach to automated camera control. In *Second Conference on Artificial Intelligence and Interactive Digital Entertainment*.

Johnston, O., and Thomas, F. 1995. *The Illusion of Life: Disney Animation*. Disney Editions.

Maybury, M. 1992. Communicative acts for explanation generation. *International Journal of Man-Machine Studies* 37(2):135–172.

Moore, J., and Paris, C. 1993. Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational Linguistics* 19(4):651–695.

Ohler, P., and Neiding, G. 1996. Cognitive modeling of suspense-inducing structures in narrative films. In Vorderer, P.; Wulff, H. J.; and Friedrichsen, M., eds., *Suspense: conceptualizations, theoretical analyses, and empirical explorations*.

Young, R. M., and Riedl, M. 2005. Integrating plan-based behavior generation with game environment. In *ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*.

Young, R. M. 2007. Story and discourse: A bipartite model of narrative generation in virtual worlds. *Interaction Studies: Social Behaviour and Communication in Biological and Artificial Systems* 8(2):177–208.

Discovering symbols from interactions — easier than explaining interactions via symbols?

Amitabha Mukerjee
Department of Computer Science and Engineering, Indian Institute of Technology, Kanpur, India

Models of symbols in logic differ fundamentally from how what symbols are in linguistic usage. Most crucially, symbols in language, in the cognitive linguistic view (Langacker 1991), are a tight binding between the phonological pole (or the label), and the semantic pole (all sorts of associations indexed to the label). In the developmental view of language, such symbols are learned based on experiential patterns. On the other hand, symbols are givens in logic, and attempts at constructing semantics, e.g. in model theory, can only define these in terms of other algebraic symbols, which are in the limit, not experientially grounded.

At one level, all algebras are autonomous axiomatic constructs, not requiring any semantics; this is why in formal logic, semantics is sometimes considered an ”illusion”. On the other hand, logic gets much of its legitimacy in terms of its being able to “formally model phenomena such as social interactions” or even in esoteric questions such as “What makes a storyline interesting” (Löwe 2009)? To address such questions in terms of formal structures is meaningful if it provides insights or holds explanatory power in understanding such interactions. In particular, it would be important for artificial systems that attempt to learn to interact.

But is logic causally prior to the interaction, or is it interaction that lets us reconstruct it in terms of logic? Arguments on both side of the debate have been made by philosophers (Quine, Fodor). I would like to focus on the semantics, and I would like to argue that to assume that the logical structures are prior to experience might call for a much larger set of assumptions (requiring us to natively know these constructs, in the Fodorian sense), and hence might be violative of Occam’s razor.

But while defining the semantics of interaction is a complex enterprise, is it really any simpler in the other direction? This would depend on the mechanism suggested for learning semantics.

We present some arguments for the naive position that semantics, in the form of what psychologists call “image schemas”, arise directly from perceptual inputs, and is thus prior to syntax. Our approach is similar to Carnap’s radical reductionism, in that we feel that semantics may form *a priori* categories based on similarity (Aufbau’s *remembered as similar* (Schilpp 1963). If this relation is represented $\mathcal{R}$, then it may, in a certain sense be a prior, though the descriptors of the sensory phenomena on which it works (“features”) may not be. In addition, our mechanism considers bottom-up attention as another prior, which selects elements from the dataset using a priori measures of relevance, thus ruling out certain combinations and limiting the set of objects to be matched by $\mathcal{R}$.

Even if the enterprise we suggest is successful, it results in a private language, which is susceptible to drift, as pointed out in late Wittgenstein. We propose that these prior internal abstractions are subsequently stabilized through interactions with a set of conventional categories that exists among others the system is interacting with. This external system serves to stabilize the units of the mental lexicon and also to modify it, by amplifying certain distinctions and merging others.

Clearly this is a forbiddingly complex argument, and the validity is subject to recognizing the mechanisms involved. Unlike Carnap’s approach however, it is possible to now construct empirical demonstrations of how the process might work. We proceed next to suggest some clues as to how this might be achieve, by presenting demonstrations from two domains. In the first, we consider a system that learns, from co-occurring language and visual sequences, relations on events. In the second, we show how an intuitive sense of optimization may constitute one of the mechanisms that lead to the lower-dimensional chunks that eventually can form the atomic units in deriving symbols, and eventually inferential systems, from a chaotic perceptual space.

Composition occurs first in semantics, and is then reflected in syntax. This then provides the basis for an *a posteriori* logic with which we can describe some of the effects in the real world. We demonstrate how simple modus ponens structures might develop as an inheritance structure for action types, and how these eventually get reflected in the syntactic constructs such as valency.

Domain 1: Actions in 2D visual input

Here we consider a simple 2D visual input, which is analyzed based on a 2D feature vector, and clustered using an unsupervised clustering algorithm, which uses euclidean

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 9–11.

distance in the feature space as its model of $\mathcal{R}$. The input and the clusters in the features space are shown in Figures 1 and 2.

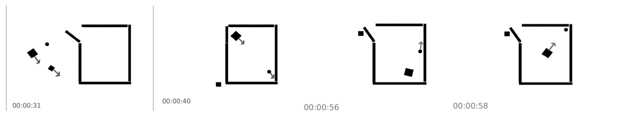


Figure 1: *Input visual sequence of relative motions between two objects*. (a) and (b) are both instances of "chase", but attentive focus (in our computational bottom-up simulation) goes to the pursuer (square) in (a) and on the leader (circle) in (b). (c) and (d) are instances of "move-away" and "move-closer". We show that the conceptual notions underlying these units correlate with four concept spaces that can be clustered in a 2D feature space representing relative position and relative velocity.

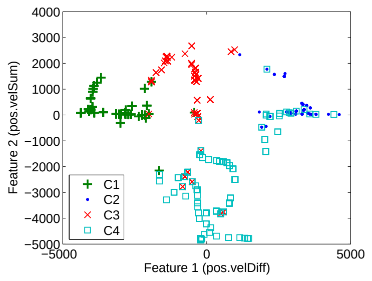


Figure 2: *Unsupervised emergence of clusters in feature space*

These four clusters map dominantly to one of the three concepts of "come closer", "move away", and "chase", but since the latter is not symmetric, we find two instances of the latter, C3 an C4, in the cluster space. These may be roughly thought of as corresponding to "Lm chases Tr" and to "Lm follows Tr" (Lm=landmark = object in attentive focus; Tr=trajector). (Satish and Mukerjee 2008)

The input to each cluster is ordered in terms of the trajector-landmark distinction. What is revealing is that if the ordering is altered, the clusters C1 and C2 continue to be recognized as the same, but clusters C3 and C4 are inverted. Thus, this discovers reflexivity for the predicates move-closer and move-away, and asymmetry for chase. In syntax, these will reflect as free word-order for the first two predicates, and for inferences a specific ordering in the latter.

Further, if the number of clusters are increased, we find preliminary evidence that the clusters move-closer and move-away are also classified into three clusters. These appear to correspond to the situations where one object (either the landmark or the trajector) is static, or where both are moving simultaneously.

Finally, these were correlated with linguistic inputs (unprocessed commentary provided by American and Indian speakers), and reasonably high correlations are obtained for these in terms of phrases in the language stream. This completes the symbol discovery process by relating the prior image schema (the semantic pole) with a label (the phonological pole). Subsequently, all relations in which this label is involved may be thought as new logical structures that can be added to the system.

Domain 2: Optimization and Symbol formation

Many input spaces are very complex and high dimensional. Thus a very large number of parameters may be required to define a padlock geometry, say. Yet, once we consider function, many of these parameters do not seem independent; we may find that as the body of the lock grows larger, the diameter of the shackle bolt also increases. Thus these two parameters are not independent. In the cognitive literature, these type of relations are called *chunks*. Discovering chunks may constitute be a fundamental step in the process of discovering symbols in high-dimensional perceptual spaces.

We consider this problem in the context of design, where function is often more formally elucidated than in everyday situations. Discovering a systematic inference structure for design problems also remains an important problem. Also, the dimensionality of many design problems (the number of parameters N required to describe a design) is very large.

In such situations, we find that optimization results in a sharp reduction of the parameter space. In general, designs are often evaluated based on multiple functional objectives, so that their optima lies on a pareto frontier; if there are d objectives, then this frontier has dimensionality $d-1$ (usually $d << N$). If the functional mapping is well-behaved, then the set of pareto designs in the N dimensional space will lie on a $d-1$ dimensional manifold. Such manifolds can be detected, and lead to the formation of chunks.

As an example, we consider an universal motor, with eight independent design parameters, optimized here for three functional aspects - mass, torque and efficiency. The resulting set of non-dominated designs lie on a 2D surface, and can be clustered in an unsupervised manner (a). The same designs are then projected from the 8 dimensional parameter space onto a 2D manifold using Locally Linear Embedding (LLE). It is seen that the mapping remains coherent, revealing reasonable inter-parameter relationships along each axis of the lower dimensional space.

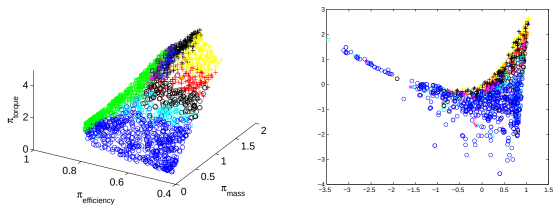


Figure 3: *Optimal surfaces form lower dimensional chunks.*

Conclusions

We have sketched the barest outline of a mechanism whereby semantic units may be learned directly, and the relations between these may give rise to syntax. When exposed to a cultural system of symbols, these proto-concepts get reified and stabilized, and rich sets of relations are added to the system.

If such a mechanism exists, it would be able to explain how logical primitives arise to begin with. The other alternative, that logical primitives are ontologically prior in some sense, faces the complex task of explaining how these set of symbols and their semantics arose in the cognitive structure. One answer may be radical nativism as in Fodor, but this requires too complex an input at the very least, and fails the Occam's razor.

While clearly not complete, this process highlights the possibility that logic may be a consequence, and not a prior, to understanding various modes of interaction. If so, the enterprise suggested in this workshop may be doomed. Logic is fine as it is in terms of a set of abstract relations. It may also be interesting as a post facto explanation of interactins etc., but the set of symbols that are used in such enterprises must be grounded in experience, and if this is to happen, the same system can also derive the rules of logic, and not the other way around.

References

Langacker, R. 1991. *Foundations of Cognitive Grammar, vol. 1*. Stanford, CA: Stanford University Press.

Löwe, B. 2009. Logic and the simulation of interaction and reasoning: Logic, computer games, and narrative. THIS BOOKLET, pp. 3–4.

Satish, G., and Mukerjee, A. 2008. Acquiring linguistic argument structure from multimodal input using attentive focus. In *7th IEEE International Conference on Development and Learning, 2008. ICDL 2008*, 43–48.

Schilpp, P. 1963. *The Philosophy of Rudolf Carnap*. Open Court Pub Co.

Speech acts as announcements

Nadine Guiraud and **Andreas Herzig** and **Emiliano Lorini**
Université Paul Sabatier, Institut de Recherche en Informatique de Toulouse (IRIT),
Logic, Interaction, Language, and Computation (LILaC), 118 route de Narbonne, 31062 Toulouse Cedex 9, France

Abstract

Our aim is to use the logic of public announcements and more generally dynamic epistemic logics as a logic of speech acts. To that end we start from a simple multimodal logic of beliefs and goals (without common belief), and add public announcements. We suppose that announcements do not modify goals. We then consider several variants of speech acts of assertive and directive force and provide a modeling in terms of speech acts.

1. Introduction

In this paper, we consider the domain of communication between agents, in particular we describe the dynamics of mental states due to communication by means of speech acts. Starting from the representation of speech acts by (Grice 1989), we consider the mental states which define the success preconditions of a speech act: intentions and belief. Thus Alice can say to Ben *I like cookies*. The mental state corresponding to this action, if Alice is honest, is that Alice *wants that Ben believes that she likes cookies*. This is the well-known assertive speech act. On this way, Grice defines other kinds of speech acts we consider in this paper: *assert*, *inform* , *confirm* , *request*, *yes-no question*. We do not only consider the mental states of the speaker, but also we represent the mental states of the hearer. In particular, we represent how the speech act influences beliefs and intentions of a locutor.

In agreement with Grice's theory, we suppose that when an agent speaks, he informs about his mental states. In the previous example, when Ben hears Alice's utterance *I like cookies*, whether he thinks Alice is honest or not, he starts to *believe that Alice wants him to beliefs that she likes cookies*. Thus, in case of a successfull communication, the hearer will understand the mental states of the speaker.

The success preconditions of speech acts are formalized in this work by means of a modal logic of beliefs and goals. Similar formalisms have been used in (Herzig and Longin 2002; Sadek 2000). The logic is extended with modal operators for announcements of Dynamic Epistemic Logic (DEL) (van Ditmarsch, van der Hoek, and Kooi 2007) in order to model the dynamics speech acts. Our formalism enables to represent how a speech act of the speaker modifies the beliefs of the hearer and so to capture the dynamic dimension of a dialogue between agents.

The paper is organized as follows: we first present the static modal logic of beliefs and goals. Then we provide a definition of model update by an announcement. Finally, we formally characterize the success preconditions of several kinds of speech acts and we develop an analysis of speech act dynamics.

2. A logic of beliefs and goals

Let $\mathrm{PRP} = \{p, q, \ldots\}$ be a countable set of propositional letters, and let $\mathrm{AGT} = \{i, j, \ldots\}$ be a countable set of agents. The set of formulas of our logic of beliefs and goals is defined by the following BNF:

$$\phi ::= p \mid \top \mid \neg\phi \mid \phi \wedge \phi \mid \mathrm{Bel}_i\phi \mid \mathrm{Goal}_i\phi$$

where p ranges over PRP and i ranges over AGT. $\mathrm{Bel}_i\phi$ reads "agent i believes that ϕ" and $\mathrm{Goal}_i\phi$ reads "agent i wants that ϕ". The others boolean operators $\bot, \vee, \rightarrow$ and $\leftrightarrow$ are defined in the standard way.

Models for that language are of the form $M = \langle W, \mathcal{B}, \mathcal{G}, V\rangle$ where

- W is a nonempty set of possible worlds;
- $\mathcal{B} : \mathrm{AGT} \longrightarrow 2^{W\times W}$ associates a belief accessibility relation to every agent;
- $\mathcal{G} : \mathrm{AGT} \longrightarrow 2^{W\times W}$ associates a goal accessibility relation to every agent;
- $V : W \longrightarrow 2^{\mathrm{PRP}}$ associates to every possible world a valuation.

We use $\mathcal{B}_i(w)$ to denote the set $\{w' \mid \langle w, w'\rangle \in \mathcal{B}_i\}$, $\mathcal{G}_i(w)$ to denote the set $\{w' \mid \langle w, w'\rangle \in \mathcal{G}_i\}$. $\mathcal{B}_i(w)$ is the set of worlds that are compatible with agent i's beliefs at world w and $\mathcal{G}_i(w)$ is the set of worlds that are compatible with agent i's goals at world w .

The truth conditions are as usual:

- $M, u \models \mathrm{Bel}_i\phi$ iff $M, v \models \phi$ for every v such that $u\mathcal{B}_i v$;
- $M, u \models \mathrm{Goal}_i\phi$ iff $M, v \models \phi$ for every v such that $u\mathcal{G}_i v$;

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 13–17.

The logic of that class of models is fairly standard. It can be axiomatized by the principles of K45 for the belief operator Bel_i, the principles of KD for the goal modal Goal_i, plus the following axiom schemas of positive and negative introspection for goals:

(1) $$\mathrm{Goal}_i\phi \rightarrow \mathrm{Bel}_i\mathrm{Goal}_i\phi$$

(2) $$\neg\mathrm{Goal}_i\phi \rightarrow \mathrm{Bel}_i\neg\mathrm{Goal}_i\phi$$

Axioms 4 and 5 for beliefs correspond together to the following semantic constraint **(C.1)** on models, Axiom D for goals corresponds to the constraint **(C.2)** and, finally, the axioms of positive and negative introspection for goals correspond together to the semantic constraint **(C.3)**.

(C.1) $w' \in \mathcal{B}_i(w)$ implies $\mathcal{B}_i(w) = \mathcal{B}_i(w')$

(C.2) $\mathcal{G}_i(w) \neq \emptyset$

(C.3) $w' \in \mathcal{B}_i(w)$ implies $\mathcal{G}_i(w) = \mathcal{G}_i(w')$

3. Adding announcements

We suppose that announcements are public, and that they only modify the agents' beliefs, but not their goals. Technically, our logic combines Kooi's version of public announcement logic (Kooi 2007) with Gerbrandy's logic of private updates (Gerbrandy 1999; Gerbrandy and Groeneveld 1997). The latter allows us to update beliefs without updating goals, and is a particular dynamic epistemic logic (Baltag, Moss, and Solecki 1998; Baltag and Moss 2004; van Ditmarsch, van der Hoek, and Kooi 2007).[1]

We extend our language by modal operators of public announcement by adding $[\phi!]\phi$ to the above BNF. The formula $[\psi!]\phi$ reads "ϕ holds after the public announcement of ψ".

In order to give semantics to announcements we define the *update* of a model by an announcement. The update of $M = \langle W, \mathcal{B}, \mathcal{G}, V\rangle$ by $\phi!$ is the model $M^{\phi!} = \langle W^{\phi!}, \mathcal{B}^{\phi!}, \mathcal{G}^{\phi!}, V^{\phi!}\rangle$ where

- $W^{\phi!} = \{u_b : u \in W\} \cup \{u_g : u \in W\}$;
- $\mathcal{B}^{\phi!} = \{\langle u_b, v_b\rangle : \langle u, v\rangle \in \mathcal{B}$ and $M, v \models \phi\} \cup \{\langle u_g, v_g\rangle : \langle u, v\rangle \in \mathcal{B}\}$;
- $\mathcal{G}^{\phi!} = \{\langle u_b, v_g\rangle : \langle u, v\rangle \in \mathcal{G}\} \cup \{\langle u_g, v_g\rangle : \langle u, v\rangle \in \mathcal{G}\}$;
- $V^{\phi!}(u_b) = V^{\phi!}(u_g) = V(u)$.

Basically, the effect of an announcement is to shrink the set of belief accessible worlds, while keeping constant the set of goal accessible worlds.

Proposition 1. *For every ϕ, if M is a model of our logic of beliefs and goals then $M^{\phi!}$ is still a model of our logic of beliefs and goals.*

Proof. The proof consists in verifying the conservation of semantic properties seen ahead.

- For **C.1**: There are two cases for belief after announcement: either $w_{2b} \in \mathcal{B}_i^{\psi!}(w_{1b})$ or $w_{2g} \in \mathcal{B}_i^{\psi!}(w_{1g})$. In each case, it implies by definition that $w_2 \in \mathcal{B}_i(w_1)$. Then $\mathcal{B}_i(w_1) = \mathcal{B}_i(w_2)$, because M satisfies constraint **C.1**. And also assume $w_3 \in \mathcal{B}_i(w_2)$. Then $w_3 \in \mathcal{B}_i(w_1)$.

  In case $w_{2b} \in \mathcal{B}_i^{\psi!}(w_{1b})$: if $w_3 \models \psi$ then $w_{3b} \in \mathcal{B}_i^{\psi!}(w_{2b})$ and $w_{3b} \in \mathcal{B}_i^{\psi!}(w_{1b})$. And if $w_{3b} \not\models \psi$ then $w_{3b} \notin \mathcal{B}_i^{\psi!}(w_{2b})$ and $w_{3b} \notin \mathcal{B}_i^{\psi!}(w_{1b})$ and $w_{3g} \notin \mathcal{B}_i^{\psi!}(w_{2b})$ and $w_{3g} \notin \mathcal{B}_i^{\psi!}(w_{1b})$.

  Then $\mathcal{B}_i^{\psi!}(w_{1b}) = \mathcal{B}_i^{\psi!}(w_{2b})$.

  In case $w_{2g} \in \mathcal{B}_i^{\psi!}(w_{1g})$: whether or not $w_3 \models \psi$ then $w_{3g} \in \mathcal{B}_i^{\psi!}(w_{2g})$ and $w_{3g} \in \mathcal{B}_i^{\psi!}(w_{1g})$.

  Then $\mathcal{B}_i^{\psi!}(w_{1g}) = \mathcal{B}_i^{\psi!}(w_{2g})$.

  Therefore, $M^{\psi!}$ satisfies constraint **C.1**.
- For **C.2**: Let us suppose $\mathcal{G}_i^{\psi!}(w_b) = \emptyset$. Then $\mathcal{G}_i(w) = \emptyset$ by the semantic definition, and it is contradictory with the constraints on M. Then $\mathcal{G}_i^{\psi!}(w_b) \neq \emptyset$. It is the same proof for $\mathcal{G}_i^{\psi!}(w_g) \neq \emptyset$. Therefore $M^{\psi!}$ satisfies constraint **C.2**.
- For **C.3**: There are two cases for belief after announcement: either $w_{2b} \in \mathcal{B}_i^{\psi!}(w_{1b})$ or $w_{2g} \in \mathcal{B}_i^{\psi!}(w_{1g})$. In each case, it implies by definition that $w_2 \in \mathcal{B}_i(w_1)$. Then $\mathcal{G}_i(w_1) = \mathcal{G}_i(w_2)$, because M satisfies constraint **C.3**. And also assume $w_3 \in \mathcal{G}_i(w_2)$. Then $w_3 \in \mathcal{G}_i(w_1)$.

  In case $w_{2b} \in \mathcal{B}_i^{\psi!}(w_{1b})$: $w_{3g} \in \mathcal{G}_i^{\psi!}(w_{2b})$ and $w_{3g} \in \mathcal{G}_i^{\psi!}(w_{1b})$. And in case $w_{2g} \in \mathcal{B}_i^{\psi!}(w_{1g})$: $w_{3g} \in \mathcal{G}_i^{\psi!}(w_{2g})$ and $w_{3g} \in \mathcal{G}_i^{\psi!}(w_{1g})$.

  Then $\mathcal{G}_i^{\psi!}(w_1) = \mathcal{G}_i^{\psi!}(w_2)$ in both cases.

  Therefore, $M^{\psi!}$ satisfies constraint **C.3**.

Items 1, 2 and 3 together imply that $M^{\psi!}$ is a model of our logic of belief and goal. □

The truth condition for announcement is:

- $M, u \models [\psi!]\phi$ iff $M^{\psi!}, u_b \models \phi$

We suppose the usual definitions of validity and satisfiability. Then the following equivalences are valid:

(3) $$[\psi!]p \leftrightarrow p \quad \text{if } p \in \mathrm{PRP}$$

(4) $$[\psi!]\neg\phi \leftrightarrow \neg[\psi!]\phi$$

(5) $$[\psi!](\phi \wedge \chi) \leftrightarrow ([\psi!]\phi \wedge [\psi!]\chi)$$

(6) $$[\psi!]\mathrm{Bel}_i\phi \leftrightarrow \mathrm{Bel}_i(\psi \rightarrow [\psi!]\phi)$$

(7) $$[\psi!]\mathrm{Goal}_i\phi \leftrightarrow \mathrm{Goal}_i\phi$$

Proof.
(3) $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]p$
iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models p$
iff $w \in V^{\psi!}(p)$
iff $w \in V(p)$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models p$.

(4) $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]\neg\varphi$
iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models \neg\varphi$

[1]For readers who are familiar with these logics: our event models have two possible events b and g, where $\mathcal{B}_i = \{\langle b, b\rangle, \langle g, g\rangle\}$ and $\mathcal{G}_i = \{\langle b, g\rangle, \langle g, g\rangle\}$.

iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \not\models \varphi$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \not\models [\psi!]\varphi$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models \neg[\psi!]\varphi$.

(5) $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!](\varphi_1 \wedge \varphi_2)$
iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models \varphi_1 \wedge \varphi_2$
iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models \varphi_1$
and $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models \varphi_2$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]\varphi_1$ and $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]\varphi_2$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]\varphi_1 \wedge [\psi!]\varphi_2$.

(6) We show that the equivalent formula

$$\neg[\psi!]\mathrm{Bel}_i\varphi \leftrightarrow \neg\mathrm{Bel}_i(\psi \to [\psi!]\varphi)$$

is valid:
$\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models \neg[\psi!]\mathrm{Bel}_i\varphi$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]\neg\mathrm{Bel}_i\varphi$, by **(4)**,
iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models \neg\mathrm{Bel}_i\varphi$
iff $\exists w'_b \in \mathcal{B}^{\psi!}(w_b)$ s.t. $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w'_b \models \neg\varphi$
iff $\exists w' \in \mathcal{B}(w)$ s.t. $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w' \models \psi$ and $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w' \models [\psi!]\neg\varphi$
iff $\exists w' \in \mathcal{B}(w)$ s.t. $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w' \models \psi \wedge \neg[\psi!]\varphi$, by **(4)**,
iff $\exists w' \in \mathcal{B}(w)$ s.t. $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w' \models \neg(\psi \to [\psi!]\varphi)$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models \neg\mathrm{Bel}_i(\psi \to [\psi!]\varphi)$.

(7) We show that the equivalent formula

$$\neg[\psi!]\mathrm{Goal}_i\varphi \leftrightarrow \neg\mathrm{Goal}_i\varphi$$

is valid:
$\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models \neg[\psi!]\mathrm{Goal}_i\varphi$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models [\psi!]\neg\mathrm{Goal}_i\varphi$, by **(4)**,
iff $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w_b \models \neg\mathrm{Goal}_i\varphi$
iff $\exists w'_g \in \mathcal{G}^{\psi!}(w_b)$ s.t. $\langle W^{\psi!}, \mathcal{B}^{\psi!}, \mathcal{G}^{\psi!}, V^{\psi!}\rangle, w'_g \models \neg\varphi$
iff $\exists w' \in \mathcal{G}(w)$ s.t. $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w' \models \neg\varphi$
iff $\langle W, \mathcal{B}, \mathcal{G}, V\rangle, w \models \neg\mathrm{Goal}_i\varphi$. □

The following rules of equivalence for announcements

(8) from $\phi \leftrightarrow \phi'$ infer $[\psi!]\phi \leftrightarrow [\psi!]\phi'$

(9) from $\psi \leftrightarrow \psi'$ infer $[\psi!]\phi \leftrightarrow [\psi'!]\phi$

preserve validity.

The above equivalences 3-7 and rules of inference 8-9 are so-called *reduction axioms*: they allow to 'move' announcement operators across the other logical operators 'inside' formulas, and to eliminate them once they face a propositional letters. Rewriting formulas starting from the innermost announcements, we can in this way eliminate announcement operators from formulas, resulting in an equivalent formula of the underlying logic of beliefs and goals.

This provides a completeness result for our logic of announcements.

Theorem 1. *Our logic of beliefs and goals is sound and complete.*

4. Speech acts as announcements

In dynamic epistemic logics announcements are usually viewed as communication actions performed by an agent that is 'outside the system', i.e. that is not part of the set of agents AGT under consideration. Our aim is to investigate how communication actions performed by agents from AGT can be modelled in our logic of announcements. To that end we consider a particular subset of announcements: we say that formulas of the form $\mathrm{Bel}_i\psi!$ or $\mathrm{Goal}_i\psi!$ are about the mental state of agent i. We identify announcements about the mental state of i, with actions performed by i. We follow Grice's idea that speech acts are expressions of intentions by the speaker, and consider that such expressions take the form $\mathrm{Goal}_i\psi!$ of announcements of goals. Such announcements of goals are directed towards an addressee.

4.1 Assertive speech acts

We consider the basic assertive as the primitive speech act and we suppose that the other kinds of speech acts are defined from this primitive speech act. We identify the expression "i *asserts* to j that ψ" with the announcement of $\mathrm{Goal}_i\mathrm{Bel}_j\psi$:

$$\mathrm{Assert}_{i,j}\psi \overset{\text{def}}{=} \mathrm{Goal}_i\mathrm{Bel}_j\psi!$$

One might express a more complex notion of assertive speech act whose content is communicative intention in the sense of Grice (Grice 1989) of the form $\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\psi$, i.e. agent i wants that j believes that i wants that j believes ψ. In the classical Gricean view of linguistic communication, a communicative intention of the speaker is aimed at the recognition by the hearer of the speaker's goal of making the hearer to believe something.

Starting from the basic assertive, we define the two speech acts *inform* and *confirm*. The speech act *inform* is a speech act *assert* under the precondition that i believes that j has no opinion about ψ. We identify it therefore with the announcement of $\mathrm{Bel}_i(\neg\mathrm{Bel}_j\psi \wedge \neg\mathrm{Bel}_j\neg\psi) \wedge \mathrm{Goal}_i\mathrm{Bel}_j\psi$:

$$\mathrm{Inform}_{i,j}\psi \overset{\text{def}}{=} \mathrm{Bel}_i(\neg\mathrm{Bel}_j\psi \wedge \neg\mathrm{Bel}_j\neg\psi) \wedge \mathrm{Goal}_i\mathrm{Bel}_j\psi!$$

The speech act *confirm* is a speech act *assert* under the precondition that i believes that j believes ψ. We identify it therefore with the announcement of $\mathrm{Bel}_i\mathrm{Bel}_j\psi \wedge \mathrm{Goal}_i\mathrm{Bel}_j\psi$:

$$\mathrm{Confirm}_{i,j}\psi \overset{\text{def}}{=} \mathrm{Bel}_i\mathrm{Bel}_j\psi \wedge \mathrm{Goal}_i\mathrm{Bel}_j\psi!$$

In both cases one might add the sincerity condition of assertives to the announcement: the speaker believes what he asserts. We then would write $\mathrm{Bel}_i\psi \wedge \mathrm{Goal}_i\mathrm{Bel}_j\psi$ instead of $\mathrm{Goal}_i\mathrm{Bel}_j\psi$ for the basic assertive of the form $\mathrm{Assert}_{i,j}\varphi$, etc.

4.2 Directive speech acts

The speech act *request* is the basic speech act of the directive kind. We define it from the assertive. We identify "i *requests* j that ψ" with "i *asserts* that he wants that j has the goal that ψ":

$$\mathrm{Request}_{i,j}\psi \overset{\text{def}}{=} \mathrm{Assert}_{i,j}\mathrm{Goal}_i\mathrm{Goal}_j\psi$$

The yes-no query "i *asks* j whether ψ" is a particular case of a request:

$$\mathrm{Ask}_{i,j}\psi \overset{\text{def}}{=} \mathrm{Request}_{i,j}(\mathrm{Bel}_i\psi \vee \mathrm{Bel}_i\neg\psi)$$

Note that if we had the sincerity condition $\mathrm{Bel}_i\psi$ for the assertive speech act, under the condition $\neg\mathrm{Bel}_i\bot$, the act of request would imply $\mathrm{Goal}_i\mathrm{Goal}_j\psi$ and the yes-no query would imply $\mathrm{Goal}_i\mathrm{Goal}_j(\mathrm{Bel}_i\psi \vee \mathrm{Bel}_i\neg\psi)$.

Suppose j is cooperative. This could be expressed by the formula

$$\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Goal}_jp \rightarrow \mathrm{Goal}_jp$$

Under this hypothesis the satisfaction conditions can be guaranteed, i.e. we have

Theorem 2.

$$\vdash [\mathrm{Request}_{i,j}p](\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Goal}_jp \rightarrow \mathrm{Goal}_jp) \rightarrow [\mathrm{Request}_{i,j}p]\mathrm{Goal}_jp$$

Proof. We have the theorems that
$\vdash [\mathrm{Request}_{i,j}p]\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Goal}_jp$
and $\vdash [\mathrm{Request}_{i,j}p]\mathrm{Goal}_jp \leftrightarrow \mathrm{Goal}_jp$

so by K $\vdash [\mathrm{Request}_{i,j}p](\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Goal}_jp \rightarrow \mathrm{Goal}_jp) \rightarrow [\mathrm{Request}_{i,j}p]\mathrm{Goal}_jp$

□

4.3 Moore sentences in speech acts

In our logic one can study Moore-like sentences for beliefs. We can prove that the assertion of $p \wedge \neg\mathrm{Bel}_ip$ leads to inconsistencies of the (higher-order) beliefs:

Theorem 3.
$\vdash \mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip) \leftrightarrow \mathrm{Bel}_i\bot$
$\vdash [\mathrm{Assert}_{i,j}\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip)!]\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot$

Proof.

- $\vdash \neg\mathrm{Bel}_ip \rightarrow \mathrm{Bel}_i\neg\mathrm{Bel}_ip$ (axiom 5)
  so $\vdash \neg\mathrm{Bel}_i\neg\mathrm{Bel}_ip \rightarrow \mathrm{Bel}_ip$
  and $\vdash \mathrm{Bel}_i\mathrm{Bel}_ip \wedge \neg\mathrm{Bel}_i\bot \rightarrow \neg\mathrm{Bel}_i\neg\mathrm{Bel}_ip$
  so $\vdash \mathrm{Bel}_i\mathrm{Bel}_ip \wedge \neg\mathrm{Bel}_i\bot \rightarrow \mathrm{Bel}_ip$
  and so $\vdash \mathrm{Bel}_i\mathrm{Bel}_ip \wedge \neg\mathrm{Bel}_i\bot \leftrightarrow \mathrm{Bel}_ip \wedge \neg\mathrm{Bel}_i\bot$
  $\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip)$
  $\leftrightarrow \mathrm{Bel}_ip \wedge \mathrm{Bel}_i\neg\mathrm{Bel}_ip \wedge (\mathrm{Bel}_i\bot \vee \neg\mathrm{Bel}_i\bot)$
  $\leftrightarrow (\mathrm{Bel}_ip \wedge \mathrm{Bel}_i\neg\mathrm{Bel}_ip \wedge \mathrm{Bel}_i\bot) \vee (\mathrm{Bel}_ip \wedge \mathrm{Bel}_i\neg\mathrm{Bel}_ip \wedge \neg\mathrm{Bel}_i\bot)$
  $\leftrightarrow \mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip \wedge \bot) \vee (\mathrm{Bel}_i\mathrm{Bel}_ip \wedge \neg\mathrm{Bel}_i\bot \wedge \mathrm{Bel}_i\neg\mathrm{Bel}_ip)$
  $\leftrightarrow \mathrm{Bel}_i\bot$
  Therefore $\vdash \mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip) \leftrightarrow \mathrm{Bel}_i\bot$
- So $[\mathrm{Assert}_{i,j}\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip)!]\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot$
  iff $\mathrm{Bel}_j(\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip) \rightarrow [\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip)!]\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot)$ by **(6)**
  iff $\mathrm{Bel}_j(\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip) \rightarrow \mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot)$ by **(7)**
  iff $\mathrm{Bel}_j(\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot \rightarrow \mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot)$
  iff $\mathrm{Bel}_j\top$
  which is a tautology.
  Therefore
  $\vdash [\mathrm{Assert}_{i,j}\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip)!]\mathrm{Bel}_j\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot$

□

On the same way, we can prove that

$\vdash [\mathrm{Assert}_{i,j}\mathrm{Bel}_i(p \wedge \neg\mathrm{Bel}_ip)!]\mathrm{Bel}_i\mathrm{Goal}_i\mathrm{Bel}_j\mathrm{Bel}_i\bot$

5. Discussion and conclusion

An immediate extension of our logic is to add announcements modifying goals, similar to (van Benthem 2007; Roy 2008; van Benthem, Girard, and Roy to appear). Such announcements would leave an agent's beliefs unaltered. In order to model this form of goal change, a product construction symmetric to the one for beliefs can be used.

Another issue we intend to investigate in the future is the theory of speech acts in the context of indirected communication. The idea is to extend the analysis presented in this paper to the case in which agents make assertions, requests, queries, etc. to other agents by using a shared blackboard which can be observed by other agents connected to it. In this case, an agent's speech act (defined in terms of announcements) modify and update the contents of the blackboard which is accessible to other agents, rather than directly modifying the beliefs and goals of other agents. In order to model this notion of blackboard, as a form of artefact used for mediating communication between agents, we intend to use the logic of acceptance recently proposed in (Lorini et al. 2009).

We also leave open the problem of integrating belief revision mechanisms into our logic, and refer to (Aucher 2003; 2008; van Ditmarsch, van der Hoek, and Kooi 2007; Baltag and Smets 2007) for possible solutions.

References

Aucher, G. 2003. A combined system for update logic and belief revision. Master's thesis, University of Amsterdam. ILLC Publications MoL-2003-03.

Aucher, G. 2008. *Perspectives on belief and change*. Ph.D. Dissertation, Université de Toulouse.

Baltag, A., and Moss, L. S. 2004. Logics for epistemic programs. *Synthese* 139(2):165–224.

Baltag, A., and Smets, S. 2007. From conditional probability to the logic of doxastic actions. In *Proc. TARK 2007*, 52–61. Presses Universitaires de Louvain.

Baltag, A.; Moss, L. S.; and Solecki, S. 1998. The logic of public announcements, common knowledge, and private suspicions. In *Proc. TARK'98*, 43–56. Morgan Kaufmann.

Gerbrandy, J., and Groeneveld, W. 1997. Reasoning about information change. *J. of Logic, Language and Information* 6(2).

Gerbrandy, J. 1999. *Bisimulations on Planet Kripke*. Ph.D. Dissertation, University of Amsterdam.

Grice, H. P. 1989. *Studies in the way of words*. USA: Harvard University Press, 3rd edition.

Herzig, A., and Longin, D. 2002. A logic of intention with cooperation principles and with assertive speech acts as communication primitives. In Castelfranchi, C., and Johnson, W. L., eds., *Proc. of the first Int. Joint Conf. on Autonomous Agent and Multi-Agent System (AAMAS 2002),* *Bologna, Italy, July 15–19*, volume 2, 920–927. Association for Computing Machinery.

Kooi, B. 2007. Expressivity and completeness for public update logic via reduction axioms. *Journal of Applied Non-Classical Logics* 17(2):231–253.

Lorini, E.; Longin, D.; Gaudou, B.; and Herzig, A. 2009. The logic of acceptance: grounding institutions on agents' attitudes. *Journal of Logic and Computation* DOI: 10.1093/logcom/exn103.

Roy, O. 2008. *Thinking before acting: intentions, logic, rational choice*. Ph.D. Dissertation, University of Amsterdam, ILLC dissertation series.

Sadek, D. 2000. Dialogue acts are rational plans. In Martin M. Taylor, F. N., and Bouwhuis, D. G., eds., *The Structure of Multimodal Dialogue II*, 167–187. Amsterdam / Philadelphia: John Benjamins.

van Benthem, J.; Girard, P.; and Roy, O. to appear. Everything else being equal: A modal logic for ceteris paribus preferences. *Journal of Philosophical Logic*.

van Benthem, J. 2007. Dynamic logic of preference upgrade. *Journal of Applied Non-Classical Logics* 17(2):157–182.

van Ditmarsch, H. P.; van der Hoek, W.; and Kooi, B. 2007. *Dynamic Epistemic Logic*. Kluwer Academic Publishers.

From generic sentences to scripts*

Lenhart Schubert
Department of Computer Science University of Rochester, PO Box 270226,
734 Computer Studies Building, Rochester, NY 14627-0226, United States of America

Abstract

One way to tackle the problem of acquiring general world knowledge to support language understanding and commonsense reasoning is to derive this knowledge by direct interpretation of general statements in ordinary language. One of several problems encountered in such an effort is that general statements frequently involve "donkey anaphora". Here a "dynamic Skolemization" approach is suggested that avoids dynamic semantics and leads naturally to script- or frame-like representations.

Introduction: Long-range goals, and the need for world knowledge

A group of us at the University of Rochester are pursuing the ambitious goal of creating a broadly knowledgable dialog agent that is motivated by curiosity, by vicarious satisfaction in fulfilling user goals, and by the sheer "pleasure" (positive utility) of engaging in conversation. However, our achievements so far fall far short of this goal: we have (1) a general knowledge representation (episodic logic, or EL) intended to match the expressiveness of human languages (e.g., (Schubert and Hwang 2000; Schubert 2000)), (2) a functioning, but still incomplete inference engine for that representation, EPILOG (e.g., (Schaeffer et al. 1993; Morbini and Schubert 2008)), (3) compositional methods for generating logical forms from English parses, (4) a large base of general "factoids" (Schubert and Van Durme 2008), and (5) a self-motivated, reward-seeking agent in a small simulated world, capable of planning ahead and simple dialogs (Liu and Schubert 2009). One of the most glaring lacunae in all of this is a sizable world knowledge base.

Two related bottlenecks, and the KNEXT system

I believe that human-like language understanding by machines is encumbered by two varieties of the notorious "knowledge acquisition bottleneck": a *pattern* acquisition bottleneck, and a *world knowledge* acquisition bottleneck. Concerning the former, I suggest that we need to reconceptualize parsing/interpretation as a massively pattern-based process, in which more specific idiomatic and semantic patterns select (and sometimes override) traditional, generic phrase structure patterns (Schubert 2009). Concerning world knowledge acquisition, we are confronted with a variety of practical and theoretical problems, some of which will be my focus here.

Our approach to knowledge acquisition is text-based, i.e., we are attempting to exploit the abundant textual materials available online nowadays. Most of our efforts so far, starting about 8 years ago, have been concerned with obtaining simple general factoids from arbitrary texts (see item (4) in the opening section). For example, the sentence "*The boy entered through the open door of the old man's shack*", when processed by our KNEXT system (e.g., (Schubert 2002; Schubert and Tong 2003; Van Durme and Schubert 2008)) yields a set of EL formulas that are automatically verbalized as

```
A BOY MAY ENTER THROUGH A DOOR OF A SHACK.
A SHACK MAY HAVE A DOOR.
A DOOR CAN BE OPEN.
A MAN MAY HAVE A SHACK.
A MAN CAN BE OLD.
```

(assuming that the intitial statistical parse is correct). We have obtained many millions of such factoids, and we regard these as providing a direct means for alleviating the pattern acquisition bottleneck; i.e., they provide patterns of predication and modification that can be used to guide a parser and semantic interpreter to produce more reliable phrasal and logical form analyses. In this way they pave the way for acquiring more complex knowledge.

Generic knowledge and donkey anaphora

In particular, with improved parsing and interpretive accuracy, we should be able to reliably interpret sentences with multiple verbs, such as "*A car crashed into a tree, killing the driver*", or "*If a car crashes into a tree, the driver is apt to be injured or killed*". Note that the first sentence is specific, but should lend itself to tentative generalization, while the second is already explicitly generic. (Generic sentences similar to this can be found, for example, in the Open Mind

*This work was supported by NSF grants IIS-0328849 and IIS-0535105.

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 19–23.

Common Sense collection (Singh et al. 2002)). A key point is that a generic conditional sentence such as the preceding one (unlike the earlier simple factoids) can enable *inferences*. For example, given a particular instance of a car crashing into a tree, we can infer that the driver of the car was probably injured or killed.

It is the acquisition of this kind of generic conditional knowledge about the world that is currently our primary concern. We are exploring a number of approaches, including abstraction from clusters of related factoids (Van Durme, Michalak, and Schubert 2009), abstraction from particular multi-verb sentences, and interpretation of available collections of generic sentences. But there are some profound difficulties afflicting all approaches that seek to derive inference-capable generic knowledge from generic sentences phrased in ordinary language:

- **lexical ambiguity**; e.g., "*Journalists write* $\underline{\textit{articles}}$" – *articles* in what sense?
- **temporal ambiguity**; e.g., "*People worry about the economy*" – now? always?
- **quantificational ambiguity**; e.g., "*Bears are* $\underline{\textit{rarely}}$ *dangerous*" – few bears, or bears in general at few times?
- **radical underspecification**; e.g., "*Cats land on their feet*" – under what conditions? and
- **donkey anaphora**; e.g., "*If a car crashes into a tree,* $\underline{\textit{the driver (of the car)}}$ *is apt to be hurt*" – which driver, which car?

There are plausible approaches to all five issues, but I will focus on the last, because it is equally important from the perspective of NL semantics and from the perspective of knowledge representation and reasoning (KR&R); my approach to donkey anaphora leads to a serendipitous convergence of these perspectives.

The problem with donkey anaphora is well-known (Geach 1962). For example, in the sentence

If a farmer buys a donkey, he rides it home,

we would like to bind the anaphoric pronouns *he* and *it* to the indefinites *a farmer* and *a donkey* in the antecedent. But if we interpret the sentence (using a generic event quantifier *Gen*) as

(Gen e: during(e,Extended-present)
[[(∃x: farmer(x))(∃y: donkey(y)) buys(x,y,e)]
→ (∃e′: after(e′,e) rides-home(x,y,e′)],

we are left with free variables in the consequent, falling outside the scopes of the existential quantifiers in the antecedent. This difficulty sparked the development of various versions of *dynamic semantics*, including discourse representation theory (DRT) (Kamp 1981; Heim 1982) and dynamic predicate logic (DPL) (Groenendijk and Stokhof 1991). In these logics, formulas are thought of as transforming an "input" variable assignment. In the above formalized sentence, the antecedent formula would transform the values of x and y, for any given value of e, so as to satisfy $\textit{buys}(x, y, e)$ (if such values exist); these values would then be available in the consequent. We say in this case that x and y are *dynamically bound* by the existential quantifiers in the antecedent, even though they lie outside the *static* scopes of those quantifiers.

A disadvantage of dynamic semantics from a KR&R perspective is loss of modularity. For example, even in a simple narrative such as

A farmer bought a donkey; he rode it home,

if we use free variables for *he* and *it* that are dynamically bound by the existential quantifiers in the first sentence, then the combined meaning of the two sentences would be lost if we interchanged their formal representations, let alone if we stored these representations separately in a KB, for independent use in inference processes. This difficulty is aggravated in extended generic contexts. For example, in the conjunctive sentence

Every man had a gun, and many used it,

the initial universal quantifier does not cover the second sentence, and consequently dynamic interpretation of the anaphoric *it* requires repetition of material from the first sentence, along the lines

Every man had a gun, and many men $\underline{\textit{had a gun}}$
and used it.

Treatment of extended generic passages in DRT was pioneered in (Carlson and Spejewski 1997), but the need to repeat earlier material for each new quantifier that applied to some portion of the generic passage was very much in evidence in that study.

Dynamic Skolemization

These problems can be avoided by Skolemizing. For example, the Skolemized representations of the two-sentence examples above would be (neglecting tense)

farmer(A), donkey(B), bought(A,B,E1);
after(E2,E1), rode-home(A,B,E2).

(∀x: man(x)) gun(G(x))∧ had(x,G(x));
(Many x: man(x)) used(x,G(x)),

where A, B, *E1*, and *E2* are Skolem constants and G is a Skolem function. Note that if we immediately Skolemize upon encountering an existentially quantified variable, the Skolem constant or function will be available for resolving subsequent anaphors.

But of course the recalcitrant cases are those involving donkey anaphora, where the existential quantifiers occur in a conditional antecedent, and thus in a negative-polarity environment. Standard Skolemization should not be expected to "work" in such a case, and at first glance, it does not. For example, consider a conditional version of the last example, tentatively Skolemized as before:

If every man has a gun, then many use it.

[(∀x: man(x)) gun(G(x))∧ have(x,G(x))]
→ (Many x: man(x)) use(x,G(x)).

This formulation has many undesirable models, e.g., models in which there are men, each has a gun, and none uses it, simply because $G(x)$ happens to denote something that is not a gun; in such a case, the conditional sentence is trivially true in virtue of the falsity of the antecedent.

But this observation, that Skolemized donkey sentences are rendered trivially true by trivially false antecedents, also holds the key to a solution: We constrain the Skolem constants or functions in the antecedent so that they cannot be trivially false. In the present example, this can be done by stipulating (separately) the following *Skolem conditional*:

$$(\forall \text{x: man(x))}[[(\exists \text{y: gun(y))have(x,y)}] \rightarrow [\text{gun(G(x))} \wedge \text{ have(x,G(x))}]].$$

Thus $G(x)$ is constrained to refer to a gun that a man x has, if indeed x is a man who has a gun; and thus if in fact every man has a gun, then the Skolemized antecedent is true, and so the conditional sentence as a whole is true only if many men x use $G(x)$, which must then be a gun that x has.

In general, we define *dynamic Skolemization* as the replacement of indefinites (as soon as they are encountered in NL interpretation) by Skolem constants or functions, while simultaneously stipulating a corresponding Skolem conditional. More formally:

Given an occurrence of a formula $(\exists y : \Phi)\Psi$, containing free variables $x_1, ..., x_m$ (and no unresolved referring expressions) in the "provisional logical form" of an English sentence being interpreted,

(a) Assert the Skolem conditional
$(\forall x_1)...(\forall x_m)[[(\exists y : \Phi)\Psi] \rightarrow [\Phi_{F(x_1,...,x_m)/y} \wedge \Psi_{F(x_1,...,x_m)/y}]]$,
where F is a new m-place function symbol;

(b) Replace the original occurrence of $(\exists y : \Phi)\Psi$ by $[\Phi_{F(x_1,...,x_m)/y} \wedge \Psi_{F(x_1,...,x_m)/y}]]$.

Some points worth noting are the following. First, as long as anaphors occur after (to the right of) their indefinite coreferents, dynamic Skolemization enables anaphora resolution "as we go". Second, we can easily generalize the above definition so that multiple indefinites occurring in sequence, and lying within the scopes of the same non-existential quantifiers, are Skolemized simultaneously. Third, the Skolem conditional can be looked at as an equivalence, since its converse is assured by existential generalization. Fifth, dynamic Skolemization reduces to ordinary Skolemization if the existential formula $(\exists y : \Phi)\Psi$ being Skolemized lies in a positive-polarity (upward-entailing) environment; i.e., in such a case the Skolem conditional can be shown to be logically redundant.

Abbreviating dynamic Skolemization: concept definitions

There is a certain verbosity in the use of dynamic Skolemization: we are writing down variants of the formula to be Skolemized, $(\exists y : \Phi)\Psi$, three times – once on each side of the Skolem conditional, and once in the Skolemized version of the original existential sentence. We can condense dynamic Skolemization through *concept definitions* that essentially name the Skolemized material, and leave the Skolem conditional itself implicit.

For example, for the earlier sentence, "*If every man has a gun, then many use it*", instead of writing down the stated Skolem conditional, we would assert

(Def HG (x) (G) [gun(G) ∧ has(x,G)]),

and rewrite the logical form of the given sentence as

[(∀x: man(x)) HG(x) → (Many x: man(x)) use(x,G(x)).

The stated *Def*-schema is a shorthand for

$$(\forall \text{x})[\text{HG(x)} \leftrightarrow [(\exists \text{y: gun(y)) have(x,y)}] \leftrightarrow [\text{gun(G(x))} \wedge \text{ have(x,G(x))}]],$$

i.e., it stipulates that *HG*(x) holds just in case x has a gun, which in turn holds just in case $G(x)$ is a gun that x has. In general, a *Def*-schema for an existential formula $(\exists y_1)...(\exists y_n)\Phi$, containing free variables $x_1, ..., x_m$,

$$(\text{Def } \Pi \ (x_1, ..., x_m) \ (F_1, ..., F_n) \ \ \Phi_{\underline{F}/\underline{y}}),$$

is shorthand for

$$(\forall x_1)...(\forall x_m)[\Pi(x_1, ..., x_m) \leftrightarrow (\exists y_1)...(\exists y_n)\Phi \ \leftrightarrow \ \Phi_{\underline{F}(x_1,...,x_m)/\underline{y}}],$$

where $\underline{F}(x_1, ..., x_m) = (F_1(x_1, ..., x_m), ..., F_n(x_1, ..., x_m))$ and $\underline{y} = (y_1, ..., y_n)$.

Creating scripts

To see how these devices lead to scripts, suppose that we are given something like the following English generic passage as a description of the process of dining at a restaurant:

Generally, when a person dines at a restaurant,
they enter it,
they get seated at a table,
they make a meal selection from a menu,
they tell the order to a waiter,
etc.

A rough provisional logical form, with anaphors subscripted for clarity with the presumed referents, is as follows:

(Gen e: (∃x: person(x))(∃y: restaurant(y)) dine-at(x,y,e))
(∃e$_1$: initial-part(e$_1$,e)) enter(x,it$_y$,e$_1$) ∧
(∃e$_2$: part-of(e$_2$,e) ∧ before(e$_1$,e$_2$))
(∃t: table(t)) get-seated-at(they$_x$,t,e$_2$) ∧
...etc...

The restrictor of the generic sentence is not a positive polarity environment, and so we Skolemize it dynamically, using a concept definition for conciseness. Note that the only free variable in the restrictor is e; for memonic reasons, we use X, Y for the Skolem functions (of e) corresponding to x and y:

(Def PDR (e) (X,Y)
[person(X) ∧ restaurant(y) ∧ dine-at(X,Y,e)]).

The body of the generic formula is a positive-polarity environment, and so we use ordinary Skolemization there. The rewritten logical form, with the anaphors resolved, thus becomes

Gen e: PDR(e))
initial-part(E$_1$(e),e) ∧ enter(X(e),Y(e),e) ∧
part-of(E$_2$(e),e) ∧ before(E$_1$(e),E$_2$(e)) ∧
table(T(e)) ∧ get-seated-at(X(e),T(e),e) ∧
...etc...

But now it is clear that the *Def*-schema for predicate *PDR*, together with the condensed logical form for the generic restaurant dining description, is remarkably script-like in structure and meaning. In fact, if we were to package the two items together, using a *Script* header, we would have

(Script PDR (e) (X,Y)
  [person(X) $\wedge$ restaurant(Y) $\wedge$ dine-at(X,Y,e)]

  initial-part(E_1(e),e) $\wedge$ enter(X(e),Y(e),e) $\wedge$
  part-of(E_2(e),e) $\wedge$ before(E_1(e),E_2(e)) $\wedge$
  table(T(e)) $\wedge$ get-seated-at(X(e),T(e),e) $\wedge$
  ...etc...),

where by definition we take the first 4 items following "*Script*" to denote (respectively) the *script name* (i.e., the predicate defined in the implicit *Def*-schema), the *script parameters*, the *script roles*, and the *script summary* (much as proposed in (Lehnert 1977)). The advantages here, however, are that

- The representation is derived directly from English logical form.
- The constituents of the script have a precise semantics, via their defined relationship to the original logical representation.
- We can refer to functional entities in scripts *outside* the scripts; this provides a solution to many problems in the interpretation of *bridging anaphora*.
- Because of the preceding point, we have complete flexibility in elaborating the derived script or variants/specializations of it.

The last two points deserve brief illustration. Suppose that we have stored the above sort of script in a KB. and we come across the sentences

John dined at Mario's. His table was wobbly.

In traditional script theory, as well as in a dynamic semantics approach, there is no simple way to equate the table referred to with the table John was seated at, if he followed the generic restaurant script. Essentially we would have to instantiate the general script, which would have John being seated at some table and eating at that table, and then identify that table with the one referred to here. In the present approach, we can simply refer to the table as $T(E)$, where E is John's dining at Mario's, and its proper role in the script instance – which need it not be made explicit – will be well-defined. The same method applies, for instance to parts of familiar kinds of objects. For example, in

John ran to a nearby house. The door was locked,

we can easily interpret *the door* as a Skolem function applied to the house in question, where that Skolem function was dynamically generated in the interpretation of a description along the lines, "*A house generally has a door at the front, through which people can enter the house*". Descriptions such as this one, which quantify over objects rather than events, can be thought of as giving rise to Minsky-like *frames* rather than scripts, upon dynamic Skolemization, and packaging with the appropriate *Def*-schema.

Concerning variant scripts and specialized scripts, consider the following elaboration of the original generic passage:

When people dine at a fancy restaurant, their table usually has a tablecloth on it.

Much as in the previous examples, we can use the available Skolem functions for this elaboration:

(Usually e: [PDR(e)$\wedge$fancy(Y(e)])
  ($\exists$c: tablecloth(c)) on(c,T(e)).

We can in turn Skolemize the tablecloth:

(Usually e: [PDR(e)$\wedge$fancy(Y(e)])
  tablecloth(C(e)) $\wedge$ on(C(e),T(e)).

Concluding remarks

I have indicated briefly that dynamic Skolemization can be used to deal systematically with donkey anaphora in generic sentences, without recourse to dynamic semantics or repetition of material. Moreover, the approach fortuitously leads to a convergence with traditional script- or frame-representations, while providing more flexibility and having a precise semantics. The connection between Skolem functions and roles in frames was noted early on in (Hayes 1981), but Hayes did not consider the possibility of Skolemizing indefinites in negative-polarity environments. Also, what has been added here is a systematic transduction from language to scripts or frames, and an elucidation of the role of Skolem constants/functions in anaphora, including donkey anaphora.

In previous work, I described both the dynamic Skolemization approach (Schubert 1999) and an approach called *implicit Skolemization* that avoids the stipulation of Skolem conditionals (Schubert 2007). But neither approach seemed fully satisfactory: Dynamic Skolemization seemed incapable of dealing with extended negative contexts where an indefinite phrase and anaphoric references to it both lie within the same downward-entailing environment; implicit Skolemization deals uniformly with all environments – but the semantics remains dynamic. However, I have recently found that negative-polarity environments can in fact be handled straightforwardly within the dynamic Skolemization approach. In essence, we distinguish *co-Skolem* constants/functions from the ordinary variety, and interpret these in terms of universal closure (whereas ordinary Skolem constants/functions are in effect interpreted in terms of existential closure). It turns out that co-Skolemization in donkey anaphora leads to "strong readings" of donkey sentences (where for instance "*If a man has a gun, he uses it*" is considered true only if multiple-gun owners use *all* their guns; in the weak readings I have restricted myself to herein, the multiple-gun owners need only use one of their guns). In current work, I am also looking at dynamic Skolemization of indefinites within attitudes such as belief, and this likewise appears feasible.

References

Carlson, G., and Spejewski, B. 1997. Generic passages. *Natural Language Semantics* 5(2):101–165.

Geach, P. 1962. *Reference and Generality*. Ithaca, N.Y.: Cornell University Press.

Groenendijk, J., and Stokhof, M. 1991. Dynamic predicate logic. *Linguistics and Philosophy* 14:39–100.

Hayes, P. 1981. The logic of frames. In Webber, B., and Nilsson, N., eds., *Readings in Artificial Intelligence*. Palo Alto, CA: Tioga Publishing Company.

Heim, I. 1982. *The Semantics of Definite and Indefinite Noun Phrases*. Ph.D. Dissertation, U. Mass.

Kamp, H. 1981. A theory of truth and semantic representation. In Groenendijk, J.; Janssen, T.; and Stokhof, M., eds., *Formal Methods in the Study of Language*. Amsterdam, Netherlands: Mathematical Centre-tracts, U. Amsterdam.

Lehnert, W. 1977. Human and computational question answering. *Cognitive Science* 1(1):47–73.

Liu, D., and Schubert, L. 2009. Incorporating planning and reasoning into a self-motivated, communicative agent. In *The 2nd Conf. on Artificial General Intelligence (AGI-09)*, 108–113.

Morbini, F., and Schubert, L. 2008. Metareasoning as an integral part of commonsense and autocognitive reasoning. In *AAAI-08 Workshop on Metareasoning*, 155–162. Forthcoming as chapter in M. T. Cox & A. Raja (eds.), *Metareasoning: Thinking about Thinking*, MIT Press.

Schaeffer, S.; Hwang, C.; de Haan, J.; and Schubert, L. 1993. EPILOG, the computational system for episodic logic: User's guide. Technical report, Dept. of Computing Science, Univ. of Alberta, Edmonton, Canada. prepared for Boeing Computer Services, Seattle, WA.

Schubert, L., and Hwang, C. 2000. Episodic Logic meets Little Red Riding Hood: A comprehensive, natural representation for language understanding. In Iwanska, L., and Shapiro, S., eds., *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language*. Menlo Park, CA, and Cambridge, MA: MIT/AAAI Press. 111–174.

Schubert, L., and Tong, M. 2003. Extracting and evaluating general world knowledge from the Brown Corpus. In *Proc. of the HLT-NAACL Workshop on Text Meaning*, 7–13.

Schubert, L., and Van Durme, B. 2008. Open extraction of general knowledge through compositional semantics. In *Abstracts of the NSF Symposium on Semantic Knowledge Discovery, Organization and Use*.

Schubert, L. 1999. Dynamic Skolemization. In Bunt, H., and Muskens, R., eds., *Computing Meaning, vol. 1*. Dortrecht: Kluwer Academic Press. 219–253.

Schubert, L. 2000. The situations we talk about. In Minker, J., ed., *Logic-Based Artificial Intelligence*. Dortrecht: Kluwer. 407–439.

Schubert, L. 2002. Can we derive general world knowledge from texts? In *Proc. of the 2nd Int. Conf. on Human Language Technology Research (HLT 2002)*, 94–97.

Schubert, L. 2007. Implicit Skolemization: Efficient reference to dependent entities. *Research on Language and Computation* 5:69–86.

Schubert, L. 2009. Language understanding as recognition and transduction of numerous overlaid patterns. In *AAAI Spring Symposium on Learning by Reading and Learning to Read*, 94–96.

Singh, P.; Lin, T.; Mueller, E. T.; Lim, G.; Perkins, T.; and Zhu, W. L. 2002. Open Mind Common Sense: Knowledge acquisition from the general public. In *Proc. of the 1st Int. Conf. on Ontologies, Databases, and Applications of Semantics for Large Scale Information Systems*.

Van Durme, B., and Schubert, L. 2008. Open knowledge extraction through compositional language processing. In *Symposium on Semantics in Systems for Text Processing (STEP 2008)*.

Van Durme, B.; Michalak, P.; and Schubert, L. 2009. Deriving generalized knowledge from corpora using WordNet abstraction. In *12th Conf. of the European Chapter of the Assoc. for Computational Linguistics (EACL-09)*.

On Basic Characteristics of Agent Types and their Influence on Rational Agent Interactions

Jan Broersen
Intelligent Systems Group, Department of Information and Computing Sciences, Universiteit Utrecht,
Postbus 80.089 3508 TB Utrecht, The Netherlands

Abstract

Logical models for agent interaction often neglect differences between agents. Yet, human agents know very well that the way they interact with other agents depends crucially on the other agent's 'type'. A good example is the game of poker, where success depends heavily on the quality of opponent models. Also, the way human agents interact with software agents is completely different from the way they interact with human agents. This paper gives a preliminary investigation into the possible game theoretic and logical dimensions of agent types that determine the ways agents interact with each other.

Introduction

In the area of logics for agent interaction and game theory, it is usually assumed that the participating agents are all of the same type. And the investigations forming an exception to this general pattern, concern limited aspects. For instance, within the BOID architecture (Broersen et al. 2002; 2001), we distinguish between between benevolent and selfish agents, and Liu (Liu 2009) distinguishes between agents with different types of memory.

However, knowing the other agents 'type' is of crucial importance when deciding on a strategy for interaction with another agent. Recent developments in AI models for the game of poker (Billings et al. 1998; Felix and Reis 2008) have acknowledged this crucial aspect of interaction. Poker is a cards game that is not so much about the strengths of players' hands, but about what opponents think is the strength of a player's hand. Different opponents have different beliefs, and managing these beliefs of opponents is of crucial importance for success in poker. Different opponent models for poker have been developed. Most are low-level opponent models, based on statistical analysis of opponent behavior. But one can also think of high level models, keeping track of, for instance, how risk averse opponents are.

For another argument in favor of opponent models, consider the well known problems with backwards induction (Aumann 1995). Examples of well-known games where backwards induction is a rational mode of reasoning are the so called centipede games (Rosenthal 1981). Example: there is a stack of 100 pennies and there are two agents. In each turn each agent can either take 1 or 2 pennies. But if an agent takes 2, the game is stopped. By backwards induction reasoning we arrive at the conclusion that the agent starting has to take 2 immediately, and thus that the game ends after the first move. However, experiments with human players have shown that in practice human agents often do not play like this. The reason is not that they are not able to see the reasoning according to backwards induction. So, apparently, there are other forms of reasoning interfering. The question is what these other forms of reasoning are. One attractive view is that agents playing the game will recognize somehow that by cooperating they can gain much more pennies. But this crucially depends on the view on the opponent in the game. Is the opponent capable of recognizing the general structure of the problem? Is the opponent even aware of the concept of cooperation? Or is the opponent a computer knowing only standard game theory (for then it will not cooperate...)? If I start by taking one penny, will my opponent be smart enough to recognize this as a *signal* for wanting to cooperate? If the opponent indeed cooperates, at what point will he then stop doing so? These are all aspects of opponent models without which the reasoning that in many cases interferes with the backwards induction reasoning, cannot take place.

It is a fundamental question then to ask what general aspects of opponent models are relevant for deciding on what strategy to employ. This brief abstract considers the differences between modeling an opponent (1) as a random chooser, (2) as a maximal expected utility decision making agent, (3) a game theoretic agent, and finally (4) an agent that is equal in all relevant decision making aspects.

An extended prisoners dilemma

I will give an extension of the standard prisoners dilemma that is designed to make it clear that our choice for what to do depends on fundamental characteristics of our model of the opponent. Table 1 gives the game. The standard prisoners dilemma is extended with one extra choice for each player: the 'be silent' action. Furthermore, 'defecting' is replaced by 'betraying'. I assume the table speaks for itself.

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 25–27.

You \ Opponent	Confess	Silent	Betray
Confess	$(4,4)$	$(0,5)$	$(0,6)$
Silent	$(5,0)$	$(3,3)$	$(1,1)$
Betray	$(6,0)$	$(1,1)$	$(2,2)$

Table 1: An extended prisoners' dilemma

Playing against a random chooser. If we assume the opponent is a random chooser we deny him any rationality. Basically, we treat the opponent not as another agent, but as part of an unpredictable environment, like in decision theory. So when deciding what to play, we can best employ the decision theoretic 'sure thing' principle (Savage 1954). The game theoretic counterpart is 'strict dominance'. The idea is that we simply look at which choice is best 'on average', without assuming the opponent has any preference for one choice over the other (after all it is a random chooser, and thus seen as a non-deterministic environmental variable). Looking at table 1 we easily reach the conclusion that we should play 'betray'. For two of the opponent 'events' (confessing and betraying), betraying is the best choice, and for one of the opponent 'events' (be silent) betraying is the second best choice. So, 'betray' is by far the best choice 'on average', which is why we should play this choice under the given assumption about the 'opponent'.

Playing against a maximal expected utility maximizer. The natural next question to ask is whether it makes a difference is we switch from seeing the other as a random chooser to seeing the other as a decision theoretic agent that sees *us* as part of *his* non-deterministic environment. The general answer to this question we leave to an extended version of this paper, but it is clear that for the particular interaction shown in the table, there is no difference: if the other reasons decision theoretically, it will choose 'betray', which makes 'betray' also the best choice for us.

Playing against a game-theoretic agent. If we assume the opponent is a game theoretic agent, we may assume it will recognize that there are two Nash equilibria. To be precise: $(2,2)$ and $(3,3)$ are Nash, while $(4,4)$ is not. Also $(2,2)$ is Pareto dominated by $(3,3)$, which in turn is Pareto dominated by $(4,4)$. Combining these two game theoretical solution concepts, we can trust the opponent to choose 'be silent' as the better option. Which is why, clearly, we also have to play 'be silent'.

Another game-theoretically inspired line of reasoning is that we can first eliminate the 'confess' choices from the table, because whatever the other agent does, 'confessing' is always the worst choice (a variant of dominance reasoning). In the table that is left, we obviously have to choose 'be silent'.

Playing against an agent that is equal in all relevant decision making aspects. I now want to consider the most fascinating category of agents: agents that are *exactly* like you. At the same time, this is the most dubious and controversial category, since it is not clear what it actually means to call two agents exactly the same and whether it makes sense to make decisions depend on it.

But we can employ a thought experiment. Assume we talk about software agents. We can call software agents the same if they run the same program. Now assume we reason from the perspective of such a software agent, and we know the opponent runs exactly the same program. Now there is a lot to say in favor of 'confess' as the best option. The point is that we simply can be very sure that the opponent reaches the same conclusion as we do, simply because it is identical to us. This draws us inevitably in the direction of the 'confess' choice.

A slightly different kind of super-rational reasoning leading to the same conclusion, is the following. If there is such a thing as an objectively (mathematically, or even empirically) best choice, than this best choice must be the same one for both players if both players are identical. But if both players necessarily play an identical choice, the best choice can only be 'confess', since $(4,4)$ is better than $(3,3)$ and $(2,2)$.

Admittedly, the arguments for playing 'confess' given in this subsection are debatable. Nevertheless, it is tempting to see the relation between the forms of reasoning associated with the different assumptions about the opponent as subsequent steps on one continuing line: if the other agent is, unlike us, completely non-intelligible (a random chooser) we *betray* it; if it is more like us and intelligible (understanding game theory, that is, also acknowledging us as an intelligible agent), we do better and *keep silent*; and only if it is exactly like us, we fully cooperate and *confess*.

Arguments favoring $(3,3)$ over $(2,2)$ *must* refer to the cognitive abilities of the opponent, since the absence of cognitive abilities on the opponent's part leads to $(2,2)$ as the best choice. But then, such arguments are about the opponent being more like us (intelligible). And taking these same arguments one step further then leads to 'confess' as the best choice.

Conclusion

Solution concepts of game theory and logics for agent interaction say nothing about the differences between agents. Implicitly they assume that all agents are the same. The main message of this abstract is however that making optimal choices in interactions with other agents depends crucially on very fundamental aspects of the opponent model.

Also, if we know what the crucial dimension of opponent models are, we know how to direct learning algorithms what to learn from observing the behavior of opponents. For instance, in iterated versions of games like the one in this abstract, agents get knowledge about their opponents through their history of moves. Also they know that their opponents get knowledge about them in the same way. One of the motivations of this work is then to find out the 'nature' of this knowledge. What are the parameters that should actually be learned is such repeated interactions in order to benefit from this knowledge in future interactions?

Many other fundamental aspects of opponents can be taken into consideration. And in a longer version of this paper, I plan to investigate other games distinguishing between these. For instance, what are the games, if any, where it makes a difference whether or not the opponent is modeled as a (1) bounded reasoner, (2) BDI-agent (3) emotional agent, (4) a moral agent?

References

Aumann, R. 1995. Backward induction and common knowledge of rationality. *Games and Economic Behavior* 8:6–19.

Billings, D.; Papp, D.; Schaeffer, J.; and Szafron, D. 1998. Opponent modeling in poker. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, 493–499. Menlo Park, CA, USA: American Association for Artificial Intelligence.

Broersen, J.; Dastani, M.; Huang, Z.; Hulstijn, J.; and van der Torre, L. 2001. An alternative classification of agent types based on BOID conflict resolution. In Kröse, B.; de Rijke, M.; Schreiber, G.; and van Someren, M., eds., *Proceedings of the 13th Belgium-Netherlands Artificial Intelligence Conference*, 79–87.

Broersen, J.; Dastani, M.; Hulstijn, J.; and van der Torre, L. 2002. Goal generation in the BOID architecture. *Cognitive Science Quarterly Journal* 2(3-4):428–447.

Felix, D., and Reis, L. P. 2008. Opponent modelling in texas hold'em poker as the key for success. In Ghallab, M.; Spyropoulos, C. D.; Fakotakis, N.; and Avouris, N. M., eds., *ECAI 2008 - 18th European Conference on Artificial Intelligence, Patras, Greece, July 21-25, 2008, Proceedings*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, 893–894. IOS Press.

Liu, F. 2009. Diversity of agents and their interaction. *J. of Logic, Lang. and Inf.* 18(1):23–53.

Rosenthal, R. 1981. Games of perfect information, predatory pricing, and the chain store. *Journal of Economic Theory* 25:92–100.

Savage, L. 1954. *The Foundations of Statistics*. New York: John Wiley and Sons.

Mathematical Modelling in TwixT

Jos Uiterwijk and **Kevin Moesker**
Department of Knowledge Engineering, Universiteit Maastricht, Postbus 616, 6200 MD Maastricht, The Netherlands

Abstract

In this paper we present results of an investigation of the connection game TwixT. This game has a very large complexity and lacks a good evaluation function based on (human) knowledge. To compensate for these drawbacks we decided to use several mathematical-modelling techniques. These are (1) board dominance based on a Voronoi tesselation, (2) shortest-path difference, (3) maximum-flow difference, and (4) shortest-path-list difference. Using these techniques we built an $\alpha\beta$ computer player. With some preliminary experiments it is shown that the mathematical-modelling tools can profitably be used as an alternative for domain knowledge, though much effort needs to be investigated to enhance the efficiency of this approach.

Game AI and Computer TwixT

Traditionally, it is considered to be a major milestone towards computer intelligence if a computer can outsmart a person in an intellectually challenging game. Claude Shannon's famous article on computer chess (Shannon 1950) laid the foundations for research on automated game playing. Building high-performance game-playing programs became a major goal of artificial intelligence research. Chess-playing programs now play at world-champion level, but they do so with limited intellectual mechanisms compared to those used by a human, substituting large amounts of computation for understanding.

For some games the best move cannot be obtained by large amounts of computation alone, and different techniques have to be tried. One of these games is TwixT, which is a two-player connection game that is invented around 1960 by Alex Randolph. TwixT challenges computer-games researchers, because the search space is large and the process of evaluating a board-position's utility value is known to be extremely complex. Little knowledge is available on how to make a computer play a strong game of TwixT. Currently no program for TwixT exists that cannot be easily beaten by an experienced TwixT player. In this paper we investigate the usability of mathematical-modelling techniques to build an artificial TwixT player.[1]

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 29–35.

The Rules of Computer TwixT

For the official rules of TwixT we refer to the article *An Introduction to TwixT* by David Bush (2000a).[2] In short, two players (White and Black) alternately make moves on a board consisting of a 24×24 square grid of holes, minus the corner holes. A move consists of placing a peg in an empty hole. The border rows exclusively are reserved for white (top and bottom) and black (left and right) pegs, respectively. After placing a peg, the same player may place a link between pegs of his color at a knight move's distance, as long as no other links are crossed. The goal for White (Black) is to connect the top and bottom (left and right) borders with a continuous chain of white (black) links. If neither side can complete such a chain, the game is a draw. Figure 1 shows a board position that is won by White and Figure 2 a board position that is a draw.

In the standard version of TwixT a player is allowed to remove links of his own color prior to placing new links. In computer TwixT this rule is replaced by the auto-linking rule, imposing that a move only consists of a peg placement and that all links that can be added to a placed peg are automatically added.

The Complexity of TwixT

In order to compare the complexity of TwixT with that of other games, we calculated two complexity measures for TwixT, i.e., its *state-space complexity* and its *game-tree complexity*.

State-Space Complexity

Allis (1994) defines the state-space complexity as: "the number of legal game positions reachable from the initial position of the game".

We calculated the number of possible peg configurations on the board, which is a lower bound on the state-space complexity of TwixT. Our calculation excludes link placement,

[1]This paper is based on the M.Sc. thesis of the second author (Moesker 2009).

[2]For a more comprehensive understanding of the game we refer to (Bush 2000b) and (Bush 2001).

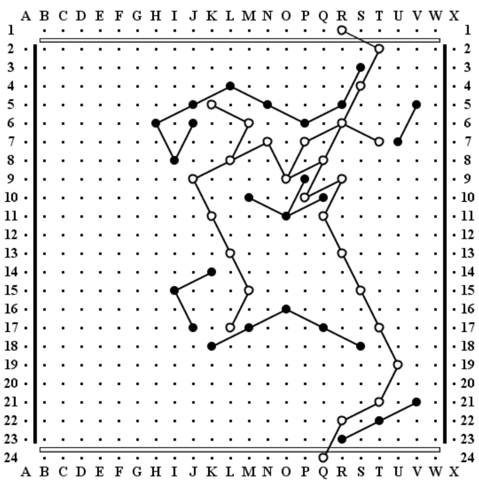


Figure 1: A TwixT position won by White.

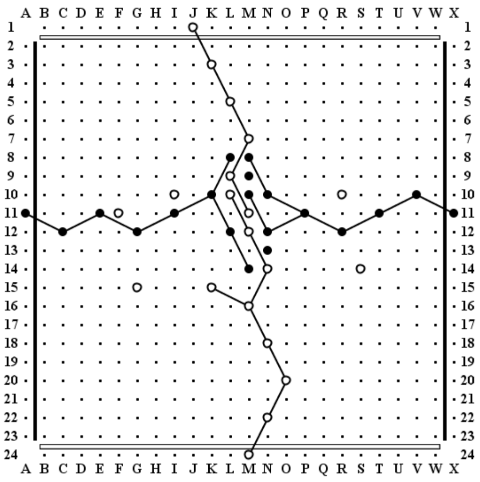


Figure 2: A drawn TwixT position.

treats mirrored board positions and rotated board positions as different positions, and includes the assumption that each player has no more than 50 pegs. Three areas exist on a TwixT board. The white border rows can only be occupied by White, the black border rows can only be occupied by Black, and the centre area can be played by both players. Each area has the corresponding capacity of 44, 44, and 484 holes.

The algorithm iteratively adds up the number of possible board positions for each total number of pegs. The summation of the calculated number of peg configurations for each number of pegs leads to a lower bound of $\approx O(10^{140})$ different board positions. Note that this is a rather crude underestimate of the real state-space complexity, since for most peg placements multiple link placements are possible, due to different sequences of peg placements. Since the calculated state-space complexity is already high, we did not find it useful to invest the effort of calculating a more precise state-space complexity.

Game-Tree Complexity

A game tree is a directed graph where nodes represent game states and arcs represent legal moves, with the root node as the initial board position. The root node of a game tree is recursively expanded for all possible continuations from each game state until states are reached where the game comes to an end. Allis (1994) defines the game-tree complexity of a game as the number of leaf nodes in the solution search tree of the initial position(s). He points out that calculating the exact game-tree complexity for a nontrivial game like chess is hardly feasible. He also mentions that $b_{average}^{d_{average}}$ gives a crude estimate of the game-tree complexity, where $b_{average}$ is an estimate on the average branching factor and $d_{average}$ is an estimate on the average depth of a game. For convenience, we assume that all pegs are placed in the $Common$ area during a game. This assumption implies that only a 22×22 area can be played, an area with a total capacity of 484.

It is tricky to estimate the average game length of TwixT, because humans generally do not play out the whole game, and because human games end quickly when the playing strength is not equal. We estimate the average TwixT game length to be 60 when both players are highly skilled and if games are played out until the end. Using an initial branching factor of 484, the final branching factor will be 424, averaging 454. This results in an estimated game-tree complexity for TwixT of $\approx O(10^{159})$.

Comparison with Other Games

Van den Herik et al. (2002) investigated the state-space and game-tree complexity of several games as game characteristics for determining solvability. Table 1 shows the identification numbers of the games, the name of the games, and the matching state-space and game-tree complexities. We have added the entry for TwixT for comparison reasons.

The estimated state-space complexity potentially exceeds all other games listed here. The estimated game-tree complexity of TwixT is above average compared to other games,

Id.	Game	ssc	gtc
1	Awari	10^{12}	10^{32}
2	Checkers	10^{21}	10^{31}
3	Chess	10^{46}	10^{123}
4	Chinese Chess	10^{48}	10^{150}
5	Connect-Four	10^{14}	10^{21}
6	Dakon-6	10^{15}	10^{33}
7	Domineering (8×8)	10^{15}	10^{27}
8	Draughts	10^{30}	10^{54}
9	Go(19×19)	10^{172}	10^{360}
10	Go-Moku (15×15)	10^{105}	10^{70}
11	Hex (11×11)	10^{57}	10^{98}
12	Kalah(6,4)	10^{13}	10^{18}
13	Nine Men's Morris	10^{10}	10^{50}
14	Othello	10^{28}	10^{58}
15	Pentominoes	10^{12}	10^{18}
16	Qubic	10^{30}	10^{34}
17	Renju (15×15)	10^{105}	10^{70}
18	Shogi	10^{71}	10^{226}
19	**TwixT**	$>> \mathbf{10^{140}}$	$\mathbf{10^{159}}$

Table 1: State-space complexities (ssc) and game-tree complexities (gtc) of various games (van den Herik et al 2002).

but it is lower than the game-tree complexity of Go. Go has an average branching factor of 250 and average game length of 150 moves, which results in a game-space complexity of approximately 10^{360} (Allis 1994). Twixt has a game-space complexity of approximately 10^{159}. The most notable difference between both games is the average game length. This is intuitive, because the shortest possible game in TwixT takes less moves than the shortest possible game in Go.

Mathematical Modelling Applied to TwixT

As a compensation for the lack of knowledge we decided to use mathematical-modelling techniques to guide a search-based AI player. We used three measures in this respect; they are (1) *board dominance*, (2) *shortest-path* difference, (3) and *maximum-flow* difference. We treat them in succession.

All the features are based on the concept of networks for both players. For each player we determine three networks, i.e., the player's *link network*, with edges corresponding with the links on the board belonging to the player; (2) the player's *allowed-link network* containing edges for each possible link for the player; and (3) the player's *combined network*, which is a superposition of the player's link and allowed-link networks. For all three networks of White we add two pegs to the position, a *source* peg *s* above the upper row, being connected to all white upper border holes, and a *target* peg *t* below the lower row, being connected to all white lower border holes. Similarly, when considering Black's networks, source and target pegs are added to the left and right of Black's border rows.

Figures 3-6 illustrate how White's perspective on a 12×12 board position is represented by White's link network, White's allowed-link network, and White's combined network.

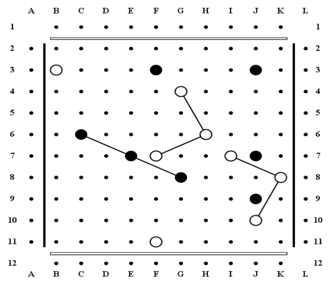


Figure 3: A 12×12 board position.

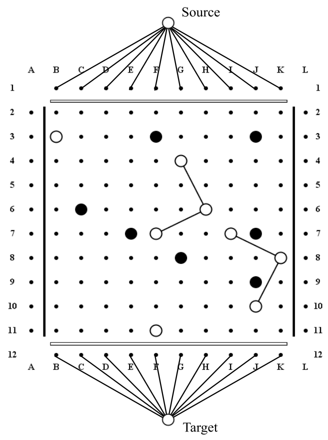


Figure 4: White's link network.

Board Dominance

To calculate board dominance, we determine the Voronoi regions of the board position. A Voronoi region represents the region of influence of a peg. *Virtual pegs* are added to a TwixT board position to correct strength-interpretation mistakes on the board. Black and white virtual pegs are placed at the corresponding border rows to indicate that the upper and lower border rows are under influence of White, and the left and right border rows are under influence of Black. Virtual pegs of corresponding color are also added at the middle of a link to indicate the strength of the link.

Figure 7 shows the Voronoi tessellation of an example 24×24 TwixT position. The white squares, at the 'upper' and 'lower' border rows, indicate White's virtual pegs and the black squares, at the 'left' and 'right' border rows, indicate Black's virtual pegs.

Subsequently, we create a Delaunay triangulation, the dual representation of a Voronoi tessellation, to model

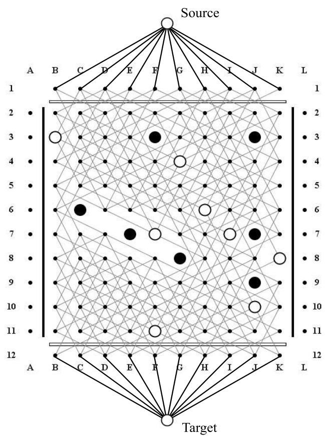


Figure 5: White's allowed-link network.

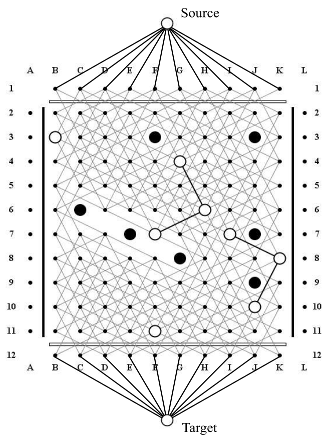


Figure 6: White's combined network.

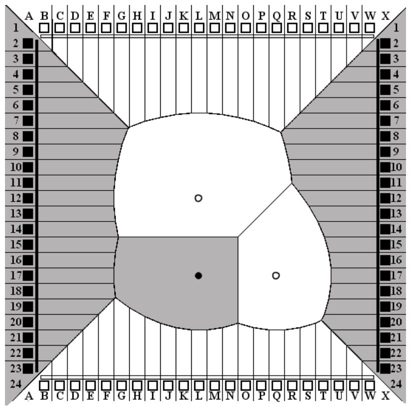


Figure 7: Voronoi representation of a TwixT position.

Voronoi-region connectedness. The dual representation is all we need for dominance checking, because all pegs with a shared Voronoi region boundary are connected in the Delaunay triangulation. Many methods exist for drawing a Delaunay triangulation from a set of points on a plane. We used Fortune's sweep algorithm (van de Berg et al. 2000). Ownership information is added to the Delaunay edges to indicate to which of the players an edge belongs. An edge is owned by White if it connects white pegs, owned by Black if it connects black pegs, and owned by no player otherwise. Figure 8 shows the Voronoi representation with the Delaunay triangulation.

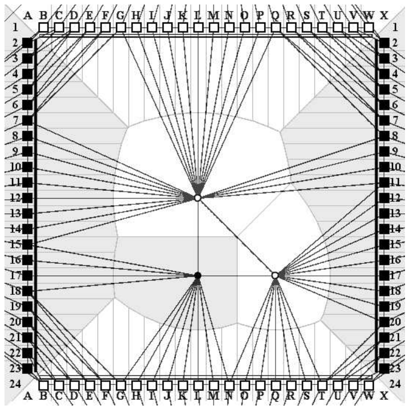


Figure 8: Voronoi representation with Delaunay triangulation.

The Delaunay triangulation has edges between pegs that are owned by White, owned by Black, and owned by both players. A close examination shows that adjacent Voronoi regions are connected in the Delaunay triangulation. Figure 9 shows the Delaunay triangulation with White's owned edges and Figure 10 shows the Delaunay triangulation with Black's owned edges. We observe that White has a continuous chain of links from the top to the border row, while Black has not. This indicates White's board dominance of the position.

Shortest-Path-Weight Feature

The shortest-path-weight feature expresses a player's minimal number of required link placements to win the game. We explain the shortest-path-weight metric from White's perspective. White's shortest-path weight is extracted from White's combined network with weight function w, with weight 0 for allowed links, weight 1 for existing links and weigh ∞ for impossible links.

The weight of a path, $p = v_1 \rightarrow v_2 \rightarrow, \ldots, v_{k-1} \rightarrow v_k$, is then defined by

$$w(p) = \sum_{i=1}^{k-1} w(v_i, v_{i+1}). \qquad (1)$$

The shortest-path weight *SP* from s to t is defined by

$$min\{w(p) : \text{ p is a path from } s \text{ to } t\}. \qquad (2)$$

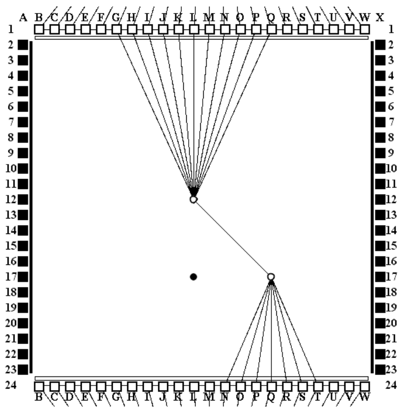


Figure 9: Delaunay triangulation with only White's owned edges.

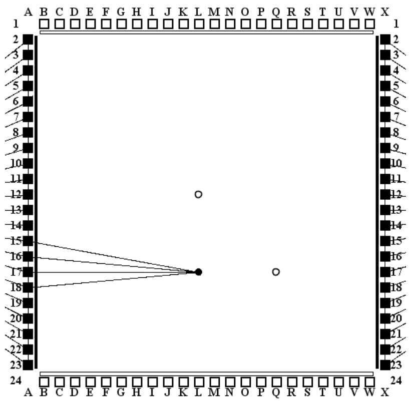


Figure 10: Delaunay triangulation with only Black's owned edges.

To determine the shortest path weight for a player we use a straightforward extension of the standard breadth-first search algorithm. The extension is needed to include the weights for edges corresponding with different types of links.

Since in some positions many moves yield the same value for the shortest-path difference, we also implemented a shortest-path-list difference *SPL*. The reasoning is that a position with multiple shortest paths is better than a position with just one such path. The implementation of this feature is a little bit complicated, and we refer to (Moesker 2009) for details.

Maximum-Flow Feature

The maximum-flow feature expresses a player's freedom to travel from side to side. "The maximum flow problem is to find a feasible flow through a single-source, single-sink flow network that is maximum" (Cormen et al. 2001). White's maximum flow is extracted from White's combined network.

The maximum flow *MF* can be found by using the Edmonds-Karp algorithm (Cormen et al. 2001). The algorithm is very straightforward. A copy of the combined network is created to serve as a residue network. Repeatedly, a shortest path from s to t is determined (using the weighting function w and extracted from the residue network until there is no path between s and t. After an augmenting path is found, all path edges with a capacity of 1 are deleted from the residue network. The path edges with infinite capacity are not deleted. The number of shortest paths found determines the maximum flow. Special caution is required concerning path selection, because different path-selection strategies might lead to a different number of augmenting paths to be found. We remove paths according to a strategy where a shortest path with vertices closest to the target is selected.

Game-Termination Feature

The game termination feature indicates if a game is a draw, a win for White, a win for Black, or still in progress. The game is a draw if both players are unable to make an uninterrupted chain of links between the corresponding sides. In terms of networks: a board position is a draw if for both players no path exists from s to t in the corresponding combined network. The existence of a path between s and t is checked with an informed depth-first search. Our 'informed' version of a DFS inserts adjacent vertices into an ordered priority queue with an increasing Euclidean distance order of vertices in the queue. Figure 11 and Figure 12 show the combined networks for White and Black, respectively, for the drawn position in Figure 2. Both figures illustrate that a drawn position leads to a failed search for a path from s to t on both White's combined network and Black's combined network.

A win for a player can be checked 'strictly' or 'softly'. For White the game is won in a strict sense if there is a path from s to t in White's link network. There is a soft win for White if White has no strict win and there is a path from s to t in White's combined link network and no such path in

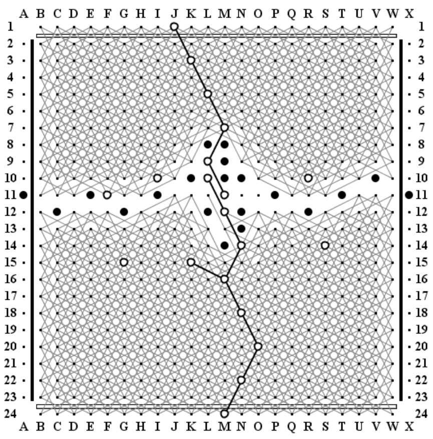


Figure 11: White's combined network for the position of Figure 2.

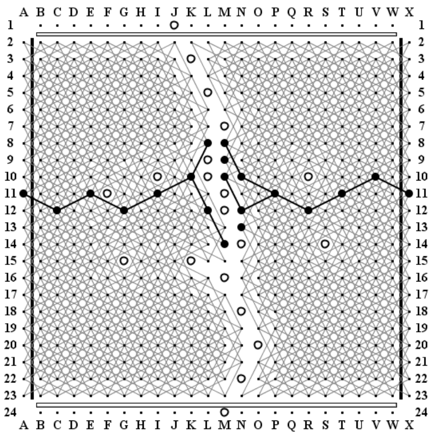


Figure 12: Black's combined network for the position of Figure 2.

Black's combined link network. A soft win does not have to be a strict win because White, the soft winning player, might not have connected to the border rows. Connecting the border rows is normally a formality but it is not always possible. There is an exceptional situation where Black, the soft losing player, cannot move because there are no legal moves left. We declare such a situation as a win for White and use the soft win condition in our AI players. An informed depth-first search based on the Euclidian distance to the target checks for possible paths between sides for White and Black. If the game is not a draw, not a win for White, and not a win for Black, then the game is still in progress.

Preliminary Experiments

To test the effectiveness of our modelling techniques, we implemented a computer player, using the standard $\alpha\beta$ algorithm (Knuth and Moore 1975) with iterative deepening. The board-dominance feature is used as a move-ordering heuristic, meaning that dominant moves are investigated before non-dominant moves. Within the categories of dominant and non-dominant moves we use the *history heuristic* (Schaeffer 1983) for further ordering. In addition, a transposition table was used (Zobrist 1970). Game termination is checked in the 'soft' sense as described above.

The evaluation function is a weighted function of the features described above. In particular:

$$Eval(p, player) = \sum_{i=1}^{3} w_i \cdot f_i(p, player) \tag{3}$$

with

$$f_1(p, player) = SP(p, player) - SP(p, opponent) \tag{4}$$

$$f_2(p, player) = MF(p, player) - MF(p, opponent) \tag{5}$$

$$f_3(p, player) = SPL(p, player) - SPL(p, opponent) \tag{6}$$

Weights are set to 10,000 for w_1 and w_2. For w_3 we used a value of 1, meaning that this feature only discriminates between moves with the same shortest-path distance and the same maximum flow. We added a randomized number between 0 and 1000 to the evaluation function to prevent the games from being deterministic. The randomized number is approximately 2% of the evaluation value range.

As a reference we implemented a basic Monte Carlo player (Bouzy and Helmstetter 2003). This player performs a one-ply search and then chooses the most promising move based on many random game simulations.

For our test we used a 8×8 board size in order to speed up the calculations. In a match of 100 games (50 times as first player, 50 times as second) with 10 minutes per player for the whole game, the $\alpha\beta$ player wins 80 games against the Monte Carlo player. Although this is by no way a conclusive result, it indicates at least that using the mathematical features within an $\alpha\beta$ framework yields promising results.

Conclusion and Future research

The preliminary experiments show that the $\alpha\beta$ player is considerably stronger than the reference Monte Carlo player. This indicates that the mathematical modelling used provides at least some compensation for the lack of domain knowledge.

Of course the experiments also reveal that the present $\alpha\beta$ player is far from competitive with strong (human) play. For future research we therefore envision at least two routes. (1) The algorithms incorporated have to be enhanced in such a way as to improve their performance in order to speed up the program considerably. (2) Due to the fact that many moves are equally good regarding their main features, shortest-path difference and maximum-flow difference, it is worthwhile to incorporate at least some basic domain knowledge (both tactical and strategic) in order to guide the search.

References

Allis, V. 1994. *Searching for Solutions in Games and Artifcial Intelligence*. Ph.D. Dissertation, Maastricht University.

Bouzy, B., and Helmstetter, B. 2003. Monte-Carlo Go Developments. In van den Herik, H.; Iida, H.; and Heinz, E., eds., *Proceedings of the 10th Advances in Computer Games Conference (ACG-10)*, 159–174. Kluwer Academic.

Bush, D. 2000a. An Introduction To TwixT. *Abstract Games Magazine* 2:9–12.

Bush, D. 2000b. TwixT Tactics, Part 1. *Abstract Games Magazine* 4:6–8.

Bush, D. 2001. TwixT Tactics, Part 2. *Abstract Games Magazine* 8:14–16.

Cormen, T.; Leiserson, C.; Rivest, R.; and Stein, S. 2001. *Introduction to Algorithms*. MIT Press and McGraw-Hill, second edition.

Knuth, D., and Moore, R. 1975. An analysis of alpha-beta pruning. *Artificial Intelligence* 6(4):293–326.

Moesker, K. 2009. TwixT: Theory, Analysis and Implementation. Master's thesis, Maastricht University.

Schaeffer, J. 1983. The history heuristic. *ICCA Journal* 6(3):16–19.

Shannon, C. 1950. Programming a Computer for Playing Chess. *Philosophical Magazine* 41:256–275.

van de Berg, B.; van Kreveld, M.; Overmars, M.; and Schwarzkopf, O. 2000. *Computational Geometry*. Springer Verlag, 2nd revised edition.

van den Herik, H.; Uiterwijk, J.; and van Rijswijck, J. 2002. Games Solved: Now and in the Future. *Artificial Intelligence* 134:277–311.

Zobrist, A. L. 1970. A New Hashing Method with Application for Game Playing. *Technical report 88*, Computer Science Department, The University of Wisconsin, Madison, WI, USA. Reprinted in (1990) *ICCA Journal* 13(2):69-73.

Emotions in Plot Generation

Rafael Pérez y Pérez
Universidad Autónoma Metropolitana, Cuajimalpa, Av. Constituyentes 1054, México D.F., Mexico

Abstract

This document describes a computer model for plot generation named MEXICA. As a distinctive characteristic, the system employs emotional links between characters and the dramatic tension of the story in progress as cue to probe memory and retrieve sequences of actions. The main claim of this document is that a story can be represented as a cluster or group of emotional links and tensions between characters that progresses over story-time; story-actions work as operators that modify such clusters. This representation provides flexibility and, by contrast with previous models, avoids the use of predefines story-structures and explicit characters goals. [1]

Introduction

MEXICA (Pérez y Pérez and Sharples 2001) is a program that generates frameworks for short stories about the Mexicas (the old inhabitants of what today is México City) based on the engagement-reflection cognitive account of writing (Sharples 1999). MEXICA does not generate natural language but sequences of actions that are coherent, interesting and novel. During engagement the system focuses on generating sequences of actions driven by content and rhetorical constraints and avoids the use of explicit goals or predefined story-structures. During reflection MEXICA evaluates the novelty and interestingness of the material produced so far and verifies the coherence of the story. Figure shows a fragment of a story developed by MEXICA.

The design of the system is based on structures known as story-actions, which are a set of actions that any character can perform in the story and whose consequences produce some change in the story-world context. The name of a story-action might be defined as single word (usually a verb) like A found B, as a couple of verbs like A followed and found B, or as a whole phrase like A followed the trace through the forest and finally found B swimming in a beautiful waterfall, where A and B represent characters in the story. MEXICA requires a dictionary of story-actions to work with. In such a dictionary the user of the system specifies the name that identifies the action, the number of characters that participate in it (maximum three actors), a set of preconditions and post conditions associated with the action, an a text (describing the action) that is employed by the system to generate its output. In this way, in MEXICA a story is defined as a sequence of story-actions. There are two types of possible preconditions and post conditions in MEXICA: 1) emotional links between characters and 2) dramatic tensions in the story.

Suddenly, virgin saw that jaguar knight had the sacred knife that had been stolen from the temple some months ago; there was not a doubt: jaguar knight was the murderer of the priest. Jaguar knight's frame of mind was very volatile and without thinking about it jaguar knight charged against virgin. Suddenly, virgin and jaguar knight were involved in a violent fight. Jaguar knight threw some dust in virgin's face. Then, using a dagger jaguar knight perforated virgin's chess. Imitating the sacred ceremony of the sacrifice, jaguar knight took virgin's heart with one hand and raised it towards the sun as a sign of respect to the gods.

Figure 1: A fragment of a story created by MEXICA.

Emotions and Tensions in MEXICA

MEXICA allows defining two types of emotional links between characters. For practical reasons all types of emotions are implemented in discrete terms with a value in the range of -3 to $+3$. Type 1 represents a continuum between love (brotherly love) and hate. Type 2 represents a continuum between being in love with (amorous love) and feeling hatred towards. For example, the action where character A falls in love with character B includes as a post condition an emotional link from A towards B of type 2 and intensity $+3$. In the same way, the action A wounds B includes as a precondition the fact that A hates B, i.e. A has an emotional link of type 1 and intensity -3 towards B. Tension is a key element in any short story. In MEXICA, it is assumed that a tension in a short story arises when a character is murdered, when the life of a character is at risk, when the health of a character is at risk (e.g., when a character has been wounded) and when a character is made a prisoner. These tensions can be defined as part of actions' post conditions and triggered when the action is performed in the story: e.g. the

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 37–39.

[1] An important part of this document has been published earlier in (Pérez y Pérez 2007).

action A wounds B triggers the post condition the health of B is at risk. In the same way, tensions can be deactivated through post conditions: e.g. the action C cures B deactivates the post condition the health of B is at risk. Notice that C cannot cure B at least B is wounded (or ill); so, the tension the health of B is at risk is a precondition of the action C cures B. There is a second group of tensions referred to as inferred tensions: 1) clashing emotions: when a character establishes two opposite emotional links towards other character; 2) love competition: when two different characters are in love with a third one; and 3) potential danger: when a character hates another character and both are located in the same place. These tensions are not defined as part of the story-actions; they are hard-coded and become active only when the emotions that trigger them are present in the story. Thus, each time an action is executed in the story in progress MEXICA verifies if inferred tensions must be triggered or deactivated. Each tension in MEXICA has associated a value. Thus, each time an action is executed the value of the tension accumulated in the tale is updated; this value is stored in a vector called Tensional Representation. The Tensional Representation records the different values of the tension over time. The Tensional Representation permits representing graphically a story in terms of the tension produced in the story. In MEXICA, a story is considered interesting when it includes increments and decrements of the tension. Figure shows the values of the tension for the fragment of the story showed in Figure .

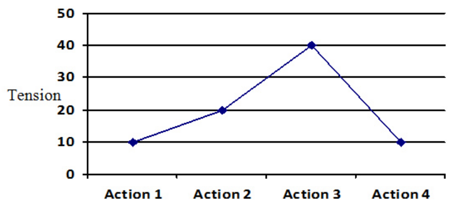
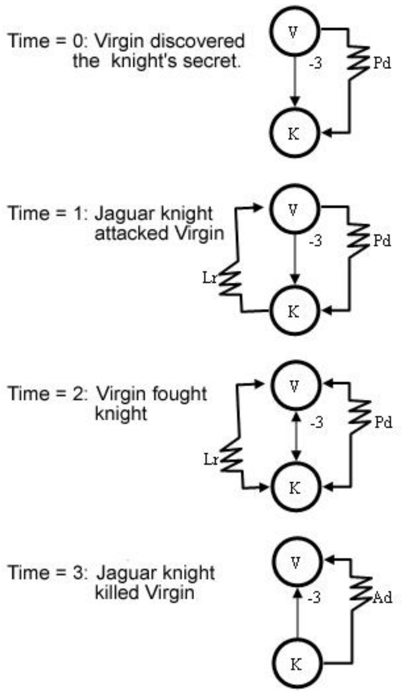


Figure 2: The tensional representation of the fragment of the story in Figure .

Discussion

The main claim of this paper is that a story can be represented as a cluster of emotional links and tensions between characters that progress over story-time (Pérez y Pérez 2007). During engagement such clusters are employed as cue to probe memory and retrieve actions to continue the story. During reflection the material produced so far is evaluated and if it is necessary modified. Figure shows the story in Figure in terms of emotional links and tensions between characters that evolve as the story progress.

In Figure , an emotional link of type 1 is represented as a solid arrow joining two characters; the intensity of the link is indicated by a signed number. An emotional link of type 2 is represented as a dashed arrow joining two characters; the intensity of the link is indicated by a signed number. Tensions between personae are represented as sharp arrows joining two characters; the type of tension is indicated by a mnemonic (Ad when an actor dies, Pd for potential danger, Lr when the life of a character is at risk, and so on).



Time = 0: Virgin discovered the knight's secret.
V
-3
Pd
K
Time = 1: Jaguar knight attacked Virgin
V
-3
Pd
Lr
K
Time = 2: Virgin fought knight
V
-3
Pd
Lr
K
Time = 3: Jaguar knight killed Virgin
V
-3
Ad
K

Figure 3: An example of how clusters of emotional links and tensions between characters evolve as the story progress.

The story develops as follows (see Figure). The first action is provided by the user. So, at $t = 0$ the virgin discovers that the knight is a murderer. The actions post conditions generate a cluster of emotional links and tensions between characters known as the story-context. As the reader can observe in Figure , at $t = 0$ the story-context shows that the virgin hates the knight because of his actions and the tension potential danger is triggered because the princess and the knight are located in the same position (by default they are located by the lake). MEXICA employs the story-context as cue to probe memory and retrieves a set of possible actions to continue the tale. One of these actions is selected at random (e.g. the knight attacks the virgin). In this way, at $t = 1$ the knight attacks the virgin putting her life at risk; so, the tension life at risk is triggered updating the story-context. The context is employed as cue to probe memory and a new action is retrieved and performed. Thus, at $t = 2$ the virgin and the knight engage in a fight. As a result the life of the knight is at risk, which produces that the knight hates the virgin. Notice that the life of the virgin continues at risk. To illustrate the fact that both characters hate each other and that the life of both characters are at risk a double headed arrow is employed to represent the emotional link and the tension Lr. Because both characters hate each other and both are located by the lake, the tension potential danger is triggered. Again, a double headed arrow is employed to represent the tension. Finally, the knight kills the virgin. This action triggers the tension actor dead; because she is dead, all virgins emotional links and tensions are deactivated although the negative emotional link from the knight towards the virgin continues active. A new search in memory is launched and the cycle continues. During reflection the system breaks impasses, and evaluates novelty and coherence. Then, it switches back to engagement and the process continues.

Conclusions

Emotions are the core elements in MEXICA: 1) All operators preconditions and post conditions are described in terms of emotional links and tensions between characters; 2) A story is represented as a cluster of emotional links and tensions that evolve over time; 3) Tension is used to evaluate the interestingness of the story produced by MEXICA; 4) During engagement MEXICA employs clusters of emotional links and tensions to progress the story. MEXCA has shown that the engagement-reflection cognitive account of writing is adequate for plot generation and that the use of emotional links and tensions between characters provides flexibility that previous programs lack (Pérez y Pérez and Sharples 2004).

References

Pérez y Pérez, R., and Sharples, M. 2001. Mexica: A computer model of a cognitive account of creative writing. *Journal of Experimental and Theoretical Artificial Intelligence* 13(2):119–139.

Pérez y Pérez, R., and Sharples, M. 2004. Three computer-based models of storytelling: Brutus, minstrel and mexica. *Knowledge Based Systems Journal* 17(1):15–29.

Pérez y Pérez, R. 2007. Employing emotions to drive plot generation in a computer-based storyteller. *Cognitive Systems Research* 8(2):89–109.

Sharples, M. 1999. *How we write*. Routledge.

Computing Quality of Life in Civilization IV[*]

Ethan Kennerly
Interactive Media, School of Cinematic Arts, University of Southern California,
University Park, LUC-310B, Los Angeles, CA 90089-2211, United States of America

Abstract

Instead of scoring the dominion of the emperor, Civilization IV can be adapted to score the quality of life of the citizens. To supplement the pedagogy of computational ethics, the author replaces the imperialist score in Civilization IV with quality of life, adapted from the medical ethics concept of a quality-adjusted life year. In order to optimize their score in the modified Civilization IV, the modern student may be motivated to compute ethics.

Introduction

In the videogame series, Civilization, the player settles an empire, advances through epochs of technology, civics, and culture. Like many imperial games, the player rules a society, determines what will be produced, which sectors of society will prosper, and what sacrifices the people will make for the glory of the empire. Decisions of sacrificing lives are naturally within the domain of ethicists (Broome 2004). However, in Sid Meier's Civilization IV, conquest and world domination are two paths to victory, wherein quality of life is sacrificed (Firaxis 2005). A player may kill thousands, sacrifice thousands, and devote a society to war in order to achieve a conquest or domination victory. These atrocities can be measured in terms of lives sacrificed (Broome 2004).

Civilization is a suitable videogame for playing with principles of history (Squire 2004). There may be a similar potential to play with computational ethics. In ethics, some simulations have refined conjectures by implementing thought experiments (Wiegel and van den Berg 2008). Civilization IV may be adapted to play with utilitarian conjectures about the moral values of a society.

Quality-Adjusted Life Years

The designers of Sid Meier's Alpha Centauri were no doubt aware of utilitarianism, as they named one of their social control technologies Ethical Calculus (Firaxis 1999). Although such a calculus is intractable, special case computations have affected many lives in the health care sector. For 30 years, some medical decision makers analyzed cost-effectiveness with a metric called a quality-adjusted life year (QALY) (Gold, Stevenson, and Fryback 2002). A QALY is composed of two factors: quality-adjustment and life years. As anyone who has ever been sick knows, health dramatically affects the satisfaction derived from living. Through subjective surveys, medical assessors measure satisfaction of living, or a quality-adjustment. The high-level formula is simple: Quality of health $\times$ years of life $=$ Individual medical well-being.

$$Q \times 1L \times Y \tag{1}$$

The conventional benchmarks for quality-adjustment are [0...1], where $0Q$ represents death and $1Q$ represents good health. For example, suppose a photographer may undergo an operation that will blind him but extend his expected life from seven years to eight years. Good health with vision ($1Q$), for seven years ($7LY$) may be computed as:

$$1Q \times 1L \times 7Y = 7QLY \tag{2}$$

The operation renders the photographer blind ($0.6Q$) and extends life to eight years ($8LY$), which may be computed as:

$$0.6Q \times 1L \times 8Y = 4.8QLY \tag{3}$$

In comparison, the photographer prefers the slightly shorter life in significantly better health. The difference is $2.2QLY$.

Even within the domain of medical ethics, this calculus is overly simplistic to account for chronic illness, multiple diseases, and extrinsic factors. Strong assumptions of risk neutrality and independence of time and quality must hold for this metric to assist in decision making (Broome 2004). Moreover, many forms of QALY do not differentiate stages of life, and do not differentiate the moral values of the persons.

Still, a quality-adjusted life year is a starting point for an ethical calculus (Broome 2004). One may extend the QALY to non-medical conditions. To accommodate mistaken beliefs, a super preference may be supposed, that being the preference the agent would have if the agent were

[*]During years of conversation, Robert Bass informed the ethics. Peter Brinson advised the user experience. The anonymous referees advised the literature. All errors and omissions are mine.

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 41–45.

well-informed (Bass 2008). This may be extended to compare between persons (Broome 2004). To distinguish this extension of the medical quality-adjusted life year, call this a well-informed, subjective, extended quality-adjusted life year (WISE-QALY). Although this does not fully define of well-being (Bass 2008), it is a convenient approximation of a good life (Broome 1999). The $0Q$ convention for death has some arithmetic advantages to compute lives not worth living. The upper limit of $1Q$, established for good health may be lifted and replaced by another benchmark. For example, it may be conventionally assigned to the average quality of life for all cities of an empire in Civilization IV that were queried during the year 2000 B.C.E.

Computing Homogenous Moral Values

The original scoring algorithm in Civilization IV was not intended to represent quality of life, but rather how glorious the civilization is, which might reflect the emperor's interests. Briefly put, Civilization IV rewards a large population, a large territory, extravagant buildings, and advanced technology (see Listing 1).

```
def calculateScore(self,argsList):
    ...
    return int(iPopulationScore + iLandScore
        + iWondersScore + iTechScore)
```

Listing 1: Computing score in original Civilization IV

An emperor could satisfy these criteria while partially ignoring the welfare of the citizens. A minimum level of citizen health and satisfaction is needed to produce wonders and research technology. Beyond this minimum, the welfare of citizens is superfluous to maximizing score.

Yet, the designers of Civilization seem to have been considering the citizens. For each city, Civilization IV abstractly represents happiness, unhappiness, health, sickness, and other parameters indicated by a quality of life survey (Stevenson and Wolfers 2008). Since satisfaction of these moral values are distinguished for each city, it would be simple to weigh them. For a baseline of indifference suppose the weighting 0 (see Table 1).

Suppose an empire of hedonists. For implementation in Civilization IV, an extreme hedonist may only value happiness and conversely disprefers unhappiness and anger. The hedonist's importance of happiness is depicted positively (0.1), of unhappiness negatively (-0.2), and of anger negatively (-0.3). These numbers only have relative meaning. Contrast them with an empire of puritans, who cares nothing for happiness (0), but values health (0.05) and conversely dislikes sickness (-0.1) (see Table 1). Many other permutations of moral values are possible, such as a temperate character, somewhat resembling Aristotle's ethical advice (see right column).

In the modified Civilization, positive moral values multiply and negative moral values divide the quality of life (see Listing 2). For convenient display, quality is represented on a scale at 100, instead of 1.0, and is an integer instead of a real number. The full code is too long to embed here. For details, you may download the modified code (Kennerly 2009) and install it into Civilization IV. This product algorithm partially normalizes the diverse ranges of scores for parameters, which have no linear correspondence to physical citizens. For example, unhappiness is incremented for each unit of population and is offset by happiness. So the value is not proportional to the population. It is an arbitrary pool of points to manage.

Moral value	Indifferent	Hedonist	Puritan	Nicomachean
happy	0	0.1	0	0.05
unhappy	0	-0.2	0	-0.1
angry	0	-0.3	0	-0.2
food	0	0	0	0
health	0	0	0.05	0.05
sick	0	0	-0.1	-0.2
eat	0	0	0	0
work	0	0	0	0.01
gold	0	0	0	0.01
science	0	0	0	0.01
culture	0	0	0	0.02
great	0	0	0	0.03

Table 1: Sample moral values to weight quality of life in Civilization IV

```
def get_quality(city_parameters, \
        moral_values):
    quality = 1
    for parameter, weight \
            in zip(city_parameters,
                moral_values):
        if 0 <= weight:
            quality *= 1 + weight \
                * float(parameter)
        else:
            quality /= 1 - weight \
                * float(parameter)
    return int(quality * 100)
```

Listing 2: Positive moral values multiply and negative moral values divide overall quality of life.

To avoid complexity of heterogenous moral values, the simplicity of Civilization IV's population and quality of life model may be exploited. Let any heterogenous population be converted into a homogenous population by averaging the moral values of the heterogenous population (see Listing 3).

```
    moral_values = aggregate_moral_values(
            (20, hedonist),
            (20, puritan),
            (60, nicomachean))
    quality = get_quality(
        py_city_parameters, moral_values)
```

Listing 3: Aggregate quality for population of 20% hedonists, 20% puritans, and 60% Nicomacheans.

Collective Utility for Homogenous Moral Values

Satisfying preferences is a popular yardstick of utility, but for conflicting interests, there is no collective solution (Arrow 1970). However, if everyone in the population is a hedonist or everyone is a puritan, Nicomachean, or any other group with similar utility functions, then aggregation is possible. A simple representation of collective well-being is the sum of individual goodness (Moulin 1991).

$$\sum_{i=1}^{n} Q_i \times 1L \times Y_i \tag{4}$$

Summing utility is not without its controversies; however, it is similar to how the original Civilization IV scores population. Individual citizens are not simulated, only city populations are, which is equivalent to adopting a utilitarian collective utility function. Therefore, aggregating well-being of city populations is already encoded in the original version of Civilization IV. In the simple model of Civilization IV's city, the total population's moral values may be the average of all its citizens.

This formula is intuitive for applying to a single person's life, but aggregating the scores of persons into a society exposes dilemmas. Suppose 100 persons are alive for one year, each with quality-adjustment of 1.

$$\sum_{i=1}^{100} 1Q \times 1L \times 1Y = 1Q \times 100L \times 1Y = 100QLY \tag{5}$$

Suppose everyone sacrifices half of their quality in order to support 150 more persons.

$$0.5Q \times 250L \times 1Y = 125QLY \tag{6}$$

This summation utility function leads to a repugnant conclusion: The highest scoring scenario would be one in which the world is saturated with lives barely worth living (Parfit 1986). The second solution posed to this problem is to average the welfare of those living.

$$\sum_{i=1}^{living} \frac{Q_i \times 1L \times Y_i}{1L \times Y_i} = \sum_{i=1}^{living} Q_i \tag{7}$$

The units of this formula are no longer QLY, but are Q. In the example, the base is now $1Q$, which is not comparable with the $100QLY$ from the summation formula.

$$\frac{1Q \times 100L \times 1Y}{100L \times 1Y} = 1Q \tag{8}$$

By averaging quality, it becomes clear that a larger populate (250) of miserable lives yields a quality ($0.5Q$).

$$\frac{0.5Q \times 250L \times 1Y}{250L \times 1Y} = 0.5Q \tag{9}$$

In Civilization IV, the lowest level of granularity is the city. Not all cities are the same size. Those with a greater population are weighted proportional to their population to retain equal importance of each living person. This is encoded in the function `aggregate_city_quality` (see Listing 4).

The living average function leads to a second repugnant conclusion. Suppose a future in which a number of persons may live. By following advice to prefer the highest average, kill all who are less happy than the average (Parfit 1986). For example, suppose 50 persons live well ($1.5Q$) and 50 persons live poorly ($0.5Q$). Their average is the same as before.

$$\frac{(1.5Q \times 50L + 0.5Q \times 50L) \times 1Y}{(50L + 50L) \times 1Y} = 1Q \tag{10}$$

If the 50 persons living poorly die, then the average quality rises:

$$\frac{1.5Q \times 50L \times 1Y}{50L \times 1Y} = 1.5Q \tag{11}$$

The second repugnant conclusion can also be illustrated in a single person's life. Suppose a person lives fifty years ($50Y$) well ($1.5Q$), and lives a further fifty years ($50Y$) poorly yet tolerably ($0.5Q$).

$$\frac{(1.5Q \times 50Y + 0.5Q \times 50Y) \times 1L}{(50Y + 50Y) \times LY} = 1Q \tag{12}$$

The person's living average can be raised by ending life after the first fifty years.

$$\frac{1.5Q \times 1L \times 50Y}{1L \times 50Y} = 1.5Q \tag{13}$$

The second repugnant conclusion can be avoided by retaining an average of those who have died. This function is similar to the average, except that those being counted is more inclusive between alternate outcomes.

$$\sum_{i=1}^{n} \frac{Q_i \times 1L \times Y_i}{1L \times E(Y_i)} \tag{14}$$

To make this function easier to interpret, those living or dead can be retained in the pool for their expected lifespan, at the time of the calculation. So, if a person at 90 years of age is expected to live to 100 years, then their loss of life is still a loss, even though they had outlived the world average life expectancy, calculated around age 1. By this formula, when the 50 less fortunate die, the average diminishes for the duration of their expected life.

$$\frac{(1.5Q \times 50L + 0Q \times 50L) \times 1Y}{(50L + 50L) \times 1Y} = 0.75Q \tag{15}$$

Likewise, the person ending their life after the first fifty (if expected to live another fifty poorly, yet tolerably).

$$\frac{(1.5Q \times 50Y + 0Q \times 50Y) \times 1L}{(50Y + 50Y) \times 1L} = 0.75Q \tag{16}$$

In Civilization IV, the lives of those dying in battle may be calculated. One could calculate for a full period, whether the person is expected to be alive or not and it would still retain the same ordering. For example, during a period of 100 years, the lives of those who live and die could continue to be averaged long after the expected duration has expired.

$$\frac{(1.5Q \times 50L \times 1Y + 0Q \times 50L \times 100Y) \times 1L}{(50L + 50L) \times 100Y} = 0.75Q \tag{17}$$

Yet calculating life-years for duration of expected living enables a mathematical shortcut. The scores of a running average can be compared between each other. This is convenient for a score that is being displayed to the player before the game is over, which is the case in Civilization IV.

This algorithm also accords with naive decision making. Consider the possible outcomes and compare them, considering the expected lives in both outcomes, rather than failing to account for a life in one possible outcome while accounting for it in alternate future.

Aggregating Over Time

One naive technique to aggregate segments of time, such as years, is to average the years. In the average quality function, the years (Y) are averaged. No further operations are necessary. When two or more segments of time are aggregated, they may be averaged by the total number of life-years, as demonstrated above. In Civilization IV, each turn represents the passage of years. The quality of life is more meaningful as a representation not of a particular turn's peak or trough, but as a running average. The lives of ancestors should be counted equally with the living. This is encoded in `average_by_years` (see Listing 4). Therefore, the value during the present turn represents an average with previous years. The information screen presents this score for quality of life. By seeing a running average, the merit of long-term strategies is incentivized.

If only snapshots are averaged, then horrible histories become incentivized. For example, if 100 persons live miserably ($0.5Q$) before 1 person lives well ($1.5Q$), then a naive snapshot seems to advise such a sacrifice.

$$\frac{0.5Q \times 100L \times 1Y}{100L \times 1Y} + \frac{1.5Q \times 1L \times 1Y}{1L \times 1Y} = 1Q \tag{18}$$

To avoid this, the summation of quality is retained and the total population is also retained, which gives more weight to generations in which a larger population live.

$$\frac{\sum_{t=1}^{T} \sum_{i=1}^{n_t} Q_i \times 1L \times Y_i}{\sum_{t=1}^{T} \sum_{i=1}^{n_t} 1L \times E(Y_i)} \tag{19}$$

We now see that many suffering for a few at a different time is not rewarded:

$$\frac{(0.5Q \times 100L \times 1Y) + (1.5Q \times 1L \times 1Y)}{(100L \times 1Y) + (1L \times 1Y)} \approx 0.51Q \tag{20}$$

As a game, this realistic running average induces boredom. In Civilization IV, the passage of time quickens. At first each turn represents 50 years. With accelerating technology, the turn duration shortens. The quality of life can be weighed by the number of years passed. In doing so, the quality score becomes stagnant. After 1000 years have passed, the effect of 10 more years is only a one percent change, so to improve the average by one percent during one turn, the current quality of life (or the current population) would have to double in the span of that turn. This is not a problem for simulation, but it is a serious problem for players to enjoy the game enough to continue playing. To avoid a stable score, and yet count that past, the past may be discounted. The default discount ratio is 1 / 256, where the current turn counts for 256 times as many years as the preceding history. A game of Civilization IV spans a few thousand years, so after 2500 years have passed, the years in the current turn are worth about 10% of the history. For details see `history_weight` in `average_by_years` (Kennerly 2009).

```
def calculateScore(self,argsList):
    city_list = \
        get_city_qaly_population_list(
            ePlayer)
    current_quality = \
        aggregate_city_quality(city_list)
    previous_scores = \
        get_previous_scores(ePlayer)
    average_quality = average_by_years(
        previous_scores, current_quality)
```

Listing 4: Computing quality of life in modified Civilization IV

Emergent Benevolence

Playing games to maximize score without this modification and with the modification reveal how the incentives change. Originally, Civilization IV incentivizes rapid growth through scoring population and territory, and rapid progress through scoring wonders and technology. The modified version, most strikingly, incentivizes minimal growth. Each new city expands the population and expands the productivity. But a new city starts without the infrastructure for health and entertainment, without aqueducts, temples, theaters, libraries, and so on. Therefore, a new city has a lower quality of life, which lowers the empire's average. Because territory is no longer a criteria for scoring, the player's score is not penalized for having a small footprint. Within a city, the option to "avoid growth" becomes attractive, as large population necessarily increases unhappiness and sickness. Growing and producing become enabling objectives of citizen well-being. The empire is less interested in conquering other territories, as the conquered territories may have lower standards of living.

Because the living conditions of the citizens is the end of the modified game, building temples, theaters and improvements is incentivized. A player may consider technologies for the benefits that they bring to their citizens, such as polytheism and monarchy to spread happiness. The player might avoid civics that detriment happiness, such as slavery. Since Civilization IV enables the acquisition of a neighboring city by expanding culture, the benevolent empire may incidentally acquire cities of its neighbors.

This modification of Civilization IV has not altered the strategies of the artificially intelligent players. All players are scored by their own citizen's quality of life. Therefore, the computer-controlled emperors remain aggressive and expansionist, even though it may lead to a lower score for their empire. Maintaining diplomacy with neighbors becomes important, as attacks impinge a city's happiness and health.

Compared to unmodified Civilization IV, scoring quality of life might lead to less imperial or fascist strategies. As a work of entertainment, intended to fuel the fantasies of their players, computing quality of life is not a requirement, and indeed guilt detracts from the make believe of total authority. Yet as an educational toy, playing with the moral values may inspire interest in computing ethics.

References

Arrow, K. J. 1970. *Social Choice and Individual Values, Second edition (Cowles Foundation Monographs Series)*. Yale University Press.

Bass, R. 2008. Subjectivism and what makes one's life better. http://www.geocities.com/amosapient/well.html.

Broome, J. 1999. *Ethics out of Economics*. Scotland: University of St Andrews.

Broome, J. 2004. *Weighing Lives*. Oxford: Oxford University Press.

Firaxis. 1999. *Sid Meier's Alpha Centauri*. Electronic Arts.

Firaxis. 2005. *Sid Meier's Civilization IV*. 2K Games.

Gold, M. R.; Stevenson, D.; and Fryback, D. G. 2002. Halys and qalys and dalys, oh my: Similarities and differences in summary measures of population health. *Annual Review of Public Health* 23:115–134.

Kennerly, E. 2009. Scoring quality of life in civilization iv. http://finegamedesign.com/qualityoflife.

Moulin, H. 1991. *Axioms of Cooperative Decision Making (Econometric Society Monographs)*. Cambridge University Press.

Parfit, D. 1986. *Reasons and Persons*. Oxford University Press, USA.

Squire, K. 2004. *Replaying History: Learning World History through playing Civilization III*. Ph.D. Dissertation, University of Wisconsin-Madison.

Stevenson, B., and Wolfers, J. 2008. Economic growth and subjective well-being: Reassessing the easterlin paradox. *Brookings Papers on Economic Activity* 2008(Spring).

Wiegel, V., and van den Berg, J. 2008. Experimental computational philosophy: shedding new lights on (old) philosophical debates. In Guerin, F.; Löwe, B.; and Vasconcelos, W., eds., *Proceedings of the AISB 2008 Symposium on Logic and the Simulation of Interaction and Reasoning*, volume 9 of *AISB 2008 Convention, Communication, Interaction and Social Intelligence, 1st-4th April 2008, University of Aberdeen*, 62–67.

Thief Belief[*]

Ethan Kennerly[1] and **Andreas Witzel**[2,3] and **Jonathan A. Zvesper**[2,3]
[1] Interactive Media, School of Cinematic Arts, University of Southern California, University Park, LUC-310B, Los Angeles, CA 90089-2211, United States of America
[2] Institute for Logic, Language and Computation, Universiteit van Amsterdam, Postbus 94242, 1090 GE Amsterdam, The Netherlands
[3] Centrum voor Wiskunde en Informatica, Science Park 123, 1098 XG Amsterdam, The Netherlands

Overview

The video game Thief: The Dark Project™ (Eidos Interactive 1998) is themed as a game of stealth, in which the player (the thief) avoids being detected by computer-simulated guards. Essentially, the player exploits the, possibly false, beliefs of the guard regarding the thief's presence. Due to the simplicity of the guard's control program, the guard's beliefs are in practice perceived as either believing that the thief is, or may be, near (having seen him or become suspicious in some other way) and acting accordingly, or not.

We conjecture that the entertainment value of a typical Thief scenario would be enhanced by a guard that acts not only depending on his own beliefs about the facts in the world, but also depending on what he believes the thief believes, including what he believes the thief believes he believes.

Besides making the interaction more realistic, we believe that dealing with, and possibly exploiting, higher-order beliefs is intrinsically enjoyable. This is not news to the logic community, in fact there exist detailed formalizations of the epistemic mechanisms involved in real-world games of knowledge and suspicion such as Cluedo (van Ditmarsch 2000). However, so far there has not been much focus on the complementary direction of putting such formalizations to work to support a computer simulation of a virtual game world.

In this paper, we will substantiate our previous discussion (Witzel, Zvesper, and Kennerly 2008) by describing an actual implementation of the pseudo-code given there and providing a detailed description of an experiment setup intended to test our conjectures about the increase in entertainment value resulting from exploiting higher-order beliefs. Before conducting the actual experiment, we are planning a small pilot study with 6 subjects in the control group and 6 subjects in the experimental group.

[*]We thank Martin Magnusson, Cédric Dégremont and three anonymous referees for discussion and helpful suggestions. The second and third authors were supported by a GLoRiClass fellowship funded by the European Commission (Early Stage Research Training Mono-Host Fellowship MEST-CT-2005-020841).

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 47–51.

Scenario

In the control scenario, when a guard has noticed a thief, the guard attacks or calls an alarm as per its scripted behavior. In the experimental scenario, the guard models the beliefs of the thief based on his own current beliefs and past observations, and acts accordingly.

In the original game, a common tactic is to shoot an arrow against a wall, in order to cause a noise which will attract the guard (who has a behavior rule along the lines of "if I hear a noise, I go there and look around"). It is questionable how innocent such a noise is, and an actually reasoning guard would probably rather conclude that it is high time to ring the alarm instead of sniffing around in the bushes. While this was likely a conscious design decision to give the player an easy and obvious way to outwit that guard, we think that a belief-based approach can provide a more flexible solution.

Instead of rigidly connecting events to resulting behaviors, we therefore introduce beliefs as an abstraction layer. The behaviors are defined depending on the beliefs, and these beliefs are modeled independently.

For example, if the guard believes that the thief is present, but the guard believes that the thief does not believe that the guard is present, the guard tries to ambush the thief, rather than attacking him openly or ringing the alarm. This and a few more behavior rules constitute the guard control program in our experimental scenario, shown in Listing 1. We will specify the events and the semantics of beliefs later on.

This approach removes from the scripter the burden of having to decide exactly which events cause what, and facilitates adjustments and more complex dependencies. For example, in our scenario a more clearly innocent noise such as breaking a twig will induce the guard to believe that there is a thief (who accidentally caused that noise), which will then trigger the behavior rule that says to ambush him.

Experiment design

We stipulate some necessary conditions for entertaining deception:

1. The exploitation of the higher-order belief is intrinsically enjoyable.

```
if Believes(guard, thief_present):
    if Believes(guard, not Believes(thief, guard_present)):
        guard.ambush(thief)
    elif Believes(guard, not Believes(thief, Believes(guard, thief_present))):
        guard.attack_or_alarm_inconspicuously()
    else:
        if not alarm_active:
            guard.alarm_quickly()
        else:
            guard.attack(thief)
```

Listing 1: Guard control program in Python style

2. The player knows there is a reward for modeling the other agents' (higher-order) beliefs.
3. The player expects the other agents to model the player's beliefs.

Based on these assumptions, we will run a small population pilot study among volunteers who are randomly assigned to a control group or experimental group.

Before play, the player is surveyed for the last video game that they played and their level of enjoyment on a five-point Likert scale, and their interest in playing that game again.

Text directly informs the player that the agents have a mental model. Without explanation, a player may fail to model the mechanisms underlying the behavior of agents in a video game. In the control scenario, the player is informed of the guard and that it will respond to noticing the thief. In the experimental scenario, the player is informed that the guard will ambush, attack, or call for help depending on how the thief behaves.

The session of play occurs on a personal computer using an animated two-dimensional prototype of the scenario. The graphical depiction and other media are identical. The setting and characters are identical. The player is also informed about the input and output conventions, which are identical for both the control group and experimental group. The difference is isolated to the behavior of the artificial agent.

During play, the player's facial expressions, vocal responses, and body language are observed. Signs of (plausibly) spontaneous emotions, such as chuckles, grins, furrowed brows, crossed legs are noted.

We conjecture that our modest belief engine encourages the original game's tactics of misleading a guard. It also encourages pretending in another way: The thief may play stupid and let himself be seen from behind by the guard, who then again thinks he can do something sneakier than ringing the alarm.

We also conjecture that our scenario encourages trepidation. A guard who has actually noticed the thief without being noticed himself will act inconspicuously while preparing a counter-attack. The blackjack is the weapon for subduing, which has a short reach. A guard that pretends to not have noticed, will swing around and stab with his sword first before the player's blackjack is in range.

Overall, we conjecture that the average enjoyment and interest in the experimental group exceed those of the control group.

To test our conjectures, after play the player is surveyed for their level of enjoyment on a five-point Likert scale, and their interest in progressing in this game. The player is also surveyed on open-ended questions about the tactics they employed, and what stood out in their experience.

Knowledge module

There are various ways in which relevant beliefs can come about: by causing or hearing noise, seeing the other one from behind, or facing each other. When scripting the behaviors, the programmer need not worry about how exactly the beliefs came about, he simply uses the familiar concept of belief in order to express the rules on a high level.

It is the job of the *knowledge module* to take care of maintaining and updating the agents' knowledge,[1] taking into account whichever events occur in the virtual world. As an agent's control script is executed, the knowledge module is queried and determines the truth value of any given formula.

We assume that the scene starts with thief and guard present and no events having occurred, and that agents cannot enter or leave. Put differently, the guard will not leave the scenario, and if the player leaves, the scenario ends. Note that this assumption is not inherent to our approach and serves mostly to avoid unnecessary complications. In the context of a computer game it is not too unnatural: Game worlds often are simulated per scene, and a scene starts and ends whenever the player enters or leaves.

In our experiment, we consider the following kinds of events:

- b_t, b_g: The thief, respectively the guard, sees the other one from behind.
- n_t, n_g: The thief, respectively the guard, makes some noise. We limit this to "relevant" events in the sense that, e.g., n_t can only occur if the thief is present.
- f: Thief and guard see each other face to face.

The epistemic effects of these events are as follows:

- b_t: The thief believes that the guard is present; the guard never assumes this event to occur, so this event does not change the guard's beliefs.

[1]In our case beliefs rather than knowledge, but we will stick with the name 'knowledge module'.

- n_t: The guard believes a thief is present; the thief believes that, if a guard is present, that guard believes that the thief is present.
- f: Thief and guard commonly believe that both are present.

In addition, these effects are commonly believed. For now, we assume that events are not forgotten. Note that a consequence of all this is that after both n_g and n_t have occurred, guard and thief commonly believe that both are present.

To model this situation formally, we use a modal logic model with history-based semantics (Fagin et al. 1995). We introduce two **propositions**, p_t and p_g, with the reading that the thief, respectively the guard, is present. A **valuation** is a function that assigns either true or false to each of these propositions, and can be denoted as a set V containing those propositions that are to be assigned true.

The set of **events**, as above, is $E = \{b_t, b_g, n_t, n_g, f\}$. A **history** H is a sequence of events. For a history H, by $Ag(H)$ we denote the set of agents involved in the events in H, i.e. $Ag(H) \subseteq \{t, g\}$, where naturally both t and g are involved in any of $\{b_t, b_g, f\}$, and only t (resp. g) is involved in n_t (resp. n_g). We also write $H - e$ for any $e \in E$ to denote the history obtained by removing all occurrences of e from H. The empty history is denoted by ϵ.

A pair (V, H) is a **state** (or **possible world**) if and only if $Ag(H) \subseteq V$. So the described initial situation is represented by the state $(\{p_t, p_g\}, \epsilon)$.

We now define accessibility relations $\dashrightarrow_t$ and $\dashrightarrow_g$ between states in our history-based model. Intuitively, $(V, H) \dashrightarrow_t (V', H')$ means that in state (V, H), t considers it possible that the state is (V', H'). According to the intuitions described above, we define these relations as follows:

$$(V, H) \dashrightarrow_t (V', H') \text{ iff } \begin{cases} t \notin V' \text{ and } H' - n_g = \epsilon & \text{if } t \notin V \\ t \in V' \text{ and } H' = H - b_g & \text{if } t \in V. \end{cases}$$

Note that in the first of these two cases, n_g is the only event which can occur in H', so the condition boils down to saying that H' may be any sequence of n_g. The second case captures the intuition that t does not assume the possibility that he is being seen from behind. The relation $\dashrightarrow_g$ is defined analogously.

We consider the doxastic language consisting of formulas of the following form:

$$\varphi ::= p_t \mid p_g \mid \neg\varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid B_t\varphi \mid B_g\varphi \ .$$

The semantics is standard relational modal semantics, see (Blackburn, de Rijke, and Venema 2001).

If we merge all doxastically equivalent states, that is, all states that make the same formulas true, our model can be depicted as in Figure 1. [2]

Note that, in contrast to our starting point from history-based semantics, this representation is static in the sense that it does not grow (or shrink) over time, as the world evolves and events occur. It is straightforwardly represented in any programming language (our implementation uses Python), and the knowledge module simply needs to keep track of which one is the current state.

Since in a computer game there is a central place where the game world is simulated, only one such model needs to be maintained, even in a scenario with more agents. Rather than maintaining a separate model inside each agent, this central model is fed with the events occurring in the game world, and queried by the agents' programs. This also means that the evaluation of formulas can be done efficiently since it corresponds to model checking, rather than testing validity as in a truly distributed setting.

Finally, note that while agents can have false beliefs, there is no need for elaborate belief revision mechanisms in our simple scenario, since there is no way for the agents to notice their false beliefs.

Technical discussion

The knowledge module we presented is a centralized implementation of the approach described in (Witzel and Zvesper 2008). The fact that it is centralized together with the simplicity of our scenario remove the necessity of finding restrictions of the doxastic language or other optimizations: About any conceivable model and implementation would be trivially tractable.

However, our scenario still shows the advantages of the modular approach which consists in introducing beliefs as an abstraction layer and separating the belief model from the agent's control program. The knowledge module can be refined independently to cope with new events or extensions, such as probabilistic beliefs, with no need to change the behavior scripts. While one could implement some ad-hoc tracking of events in the agent control program, this is error-prone and difficult to maintain, as already in our small test scenario the exact dependencies on events are getting confusing: What does "if n_t or b_g have occurred, but neither n_g nor f have, then. . . " mean?

Of course a knowledge module may also need debugging, but this can be done separately from the rest of the program. To this end, it would be interesting to define a *Why*() function, which should give a (useful) explanation of why a given belief formula holds in a given state.

One extension that is easy to incorporate into our knowledge module are decaying events, or unstable states: If the guard makes some noise, like whistling (which he obviously would only do if he does not believe a thief is present), he might have forgotten about that when he 10 minutes later sees the thief from behind, and therefore *not* conclude that the thief believes the guard is present. This could be modeled by a spontaneous transition from state 5 to 2 after some time.

The advantages of our finite-state-machine-like representation is that it is straightforwardly represented using a very simple data structure, and that it is a static, pre-computed model, so there is no expense at run-time and no surprises from uncontrolled growth. However, for bigger scenarios, it may not be feasible to keep the whole model in memory all the time. Mechanisms to generate the model only locally as

[2]This is the so-called 'bisimulation contraction' of our original model. Note that in our case, if (V, H) and (V', H') are doxastically equivalent, so are (V, He) and $(V', H'e)$ for any event e.

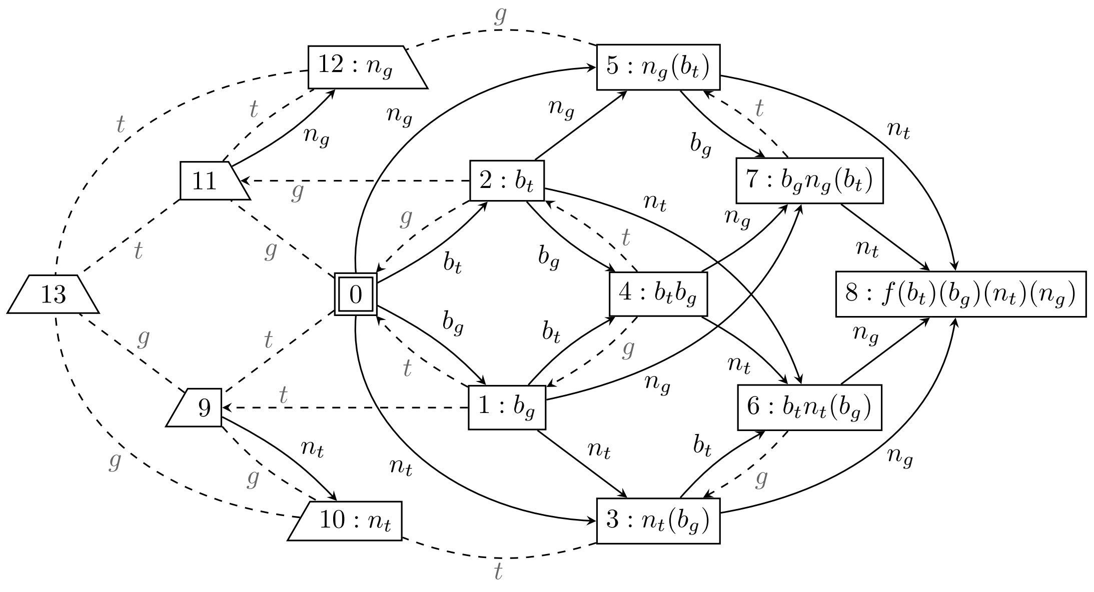


Figure 1: Representation of our doxastic model. The numbered nodes represent states and are annotated with one of the equivalent possibilities of what events have (optionally) taken place. Boxed states have $V = \{p_t, p_g\}$, a missing left corner means $p_g \notin V$, and a missing right corner means $p_t \notin V$. State 0 represents the initial situation. Solid arrows with black labels show event transitions, dashed arrows and edges (the latter corresponding to bidirectional arrows) with gray labels show accessibility relations. The following are omitted for clarity: reflexive transitions for each optional event at each state; f-transitions from each boxed state to 8; and reflexive accessibilities, except for g in states 2 and 6 and for t in states 1 and 7.

needed, and strategies for expanding, shrinking, or swapping out unused parts of the model may be necessary.

A very promising alternative is Dynamic Epistemic Logic (DEL). Instead of using one model to represent all possible ways in which the world may evolve, one starts with a model representing only the initial situation. Events are represented as so-called 'update models', which are applied to the current model as the events occur, changing the model to reflect the resulting situation.

It is then desirable to ensure that the model will not grow indefinitely. In our scenario, this can be achieved by showing that the events are commutative and idempotent in an adequate sense, and that the update mechanism preserves minimality with respect to bisimulation. It would be interesting to look at such properties of DEL models and events in a more general way.

Besides this 'online' use, DEL could also be used simply as a description language for initial situation and events, from which a static model similar to ours could be precomputed 'offline'. Again, considerations about the size of the resulting model, or language restrictions to ensure certain complexity bounds, would be relevant.

An interesting suggestion that would also simplify reasoning is to consider so-called 'interaction axioms' between the belief modalities of the players. For example, in (Lomuscio and Ryan 1998) the authors show that (2WD) is valid on two-player 'hypercubes', i.e., models that are based on the Cartesian product of an interpreted system's state space:

$$\Diamond_1\Box_2 p \implies \Box_2\Diamond_1 p. \qquad \text{(2WD)}$$

In a different context (van Ditmarsch and Labuschagne 2007) consider interaction axioms that characterise different Theories of Mind, including 'autistic' and 'deranged'. It is an interesting question whether interaction axioms that capture agents that are *fun* to interact with might exist.

Another point worth noting is that we do not distinguish between *knowledge* and *true belief*. There are many philosophical discussions concerning differences between these two notions, and formally they are generally taken to be different as well. Beliefs should really be *revisable* for example, which is something we have not considered here. A possible future direction of research would be to use models and languages that take both belief and knowledge into account, for example along the lines of (Shoham and Leyton-Brown 2009, Chapter 13). Some actions would then generate knowledge, while some would (only) generate belief. This corresponds to the distinction van Benthem (2007) draws between 'hard' and 'soft' information. For example, if you *see* something then you might be said to know it, but if somebody you don't entirely trust tells you something then you might only believe it.

Finally, it may be desirable to get rid of manually designed behavior rules altogether, and let the agents act purely based on their beliefs and some abstractly defined goals. In order to accomplish this, a very promising approach is to combine a game world that uses deductive planning, e.g. (Magnusson and Doherty 2008), with a belief model such as the one we discussed. In this way, the planning agents would be enabled to reason about the epistemic preconditions and consequences of their actions and the epistemic parts of their goals.

A very long-term vision is then to have an artificial guard that by himself adopts behaviors similar to the ones we described, or to the tactics we conjectured for the human thief.

References

Blackburn, P.; de Rijke, M.; and Venema, Y. 2001. *Modal Logic*. Cambridge University Press.

Eidos Interactive. 1998. Thief: The Dark Project. `http://www.eidosinteractive.com/games/info.html?gmid=34`.

Fagin, R.; Halpern, J. Y.; Vardi, M. Y.; and Moses, Y. 1995. *Reasoning about knowledge*. MIT Press.

Lomuscio, A., and Ryan, M. 1998. Ideal agents sharing (some!) knowledge. In *Proceedings of Proceedings of the 13th European Conference on Artificial Intelligence*, 557–561. John Wiley & sons.

Magnusson, M., and Doherty, P. 2008. Logical agents for language and action. In *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE-08)*.

Shoham, Y., and Leyton-Brown, K. 2009. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press.

van Benthem, J. 2007. Dynamic logic for belief revision. *Journal of Applied Non-Classical Logic* 17(2):129–155.

van Ditmarsch, H. P., and Labuschagne, W. A. 2007. My beliefs about your beliefs: a case study in theory of mind and epistemic logic. *Synthese* 155(2):191–209.

van Ditmarsch, H. P. 2000. *Knowledge games*. Ph.D. Dissertation, Groningen University. ILLC Dissertation Series 2000-06.

Witzel, A., and Zvesper, J. A. 2008. Epistemic logic and explicit knowledge in distributed programming (short paper). In *Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2008)*.

Witzel, A.; Zvesper, J. A.; and Kennerly, E. 2008. Explicit knowledge programming for computer games. In *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE-08)*.

Logical Agents that Plan, Execute, and Monitor Communication*

Martin Magnusson and **David Landén** and **Patrick Doherty**
Department of Computer and Information Science, Linköping University, Sweden

Abstract

Real and simulated worlds that involve cooperation between several autonomous agents create highly dynamic conditions. An individual agent can not hope to rely on fixed behaviors, but needs to plan both physical actions and communication actions that enlist other agents to help achieve the goal. We present a logical agent that plans regular actions and speech acts, monitors their execution, and recovers from unexpected failures. The architecture is based on automated reasoning in a formal logic and is grounded in a physical robot platform in the form of an autonomous helicopter. We apply the system to a knowledge gathering goal and illustrate how it deals with the complexities of the task.

Introduction

Imagine the chaotic aftermath of a natural disaster. Teams of rescue workers search the afflicted area for people in need of help, but they are hopelessly understaffed and time is short. Fortunately, they are aided by a small fleet of unmanned aerial vehicles (UAVs) that can be requested to carry out tasks autonomously. The UAVs help quickly locate injured by scanning large parts of the area from above using infrared cameras and communicating the information to the command and control center (CCC) in charge of the emergency relief operation.

Complex tasks like these occur frequently in real world situations as well as in training simulations and computer games. Achieving their goals requires the cooperation of many independent agents. Each agent faces an environment that can change in response to actions by other agents in addition to its own. It is unrealistic to assume that such an agent could be made autonomous by equipping it with fixed behaviours for all possible situations that might arise. Instead, the agent must automatically construct plans of action adapted to the situation at hand. These multi-agent plans involve other agents' mental states and communicative acts to affect them. In addition, assumptions made during planning must be monitored during execution so that the agent can autonomously recover from failures, which are prone to happen in any dynamic environment. These are significant challenges for any proposed planning formalism.

We extend Temporal Action Logic (Doherty and Kvarnström 2007), a first-order language with a well-developed methodology for representing actions that has previously been used for planning (Magnusson 2007), with syntactic operators that express the modalities of belief and commitment. This makes it possible to formalize *inform* and *request* speech acts, and to plan both physical and communicative actions in a multi-agent setting. Automation is provided by a natural deduction theorem prover that integrates planning, execution, monitoring, and plan revision. The resulting physical actions are executed (in simulation) by a delegation framework (Doherty and Meyer 2007), built using the Java agent development framework[1] (JADE), and communicative actions are realized as standardized FIPA ACL (Foundation for Intelligent Physical Agents 2002) speech acts. We describe this agent architecture and its use in solving the above scenario.

Related Work

Early planning research tended to ignore any processes that occur before or after the planning phase, and many classical planning algorithms depended crucially on a global closed world assumption. Agent programming languages provide interesting alternatives that often incorporate execution and monitoring, but move planning to a less prominent role or eliminate planning altogether, which might result in some sacrifice of flexibility in unforeseen situations.

Modern planning research tries to relax the classical restrictions while keeping the planning process in focus. Shanahan's agent architecture (Shanahan 2000) based on the Event Calculus seamlessly integrates planning and plan revision. Though it has not been applied in multi-agent settings, and its implementation as an abductive logic program is not sufficiently expressive to represent communicative acts conveniently.

Searle's *speech acts* (1969) characterize natural language utterances as actions with conditions upon their execution and effects on the mental states of others. Speech acts form a basis for expressing communicative acts as planning operators, as is done by Perrault, Allen, and Cohen (1978). They generate plans that involve both regular actions and speech acts, but the implementation uses limited versions of speech acts in the form of STRIPS-like planning operators.

*This work is supported in part by the Swedish Foundation for Strategic Research (SSF) Strategic Research Center MOVIII, the Swedish Research Council Linnaeus Center CADICS, and CENIIT, the Center for Industrial Information Technology.

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 53–58.

[1]http://jade.tilab.com/

Speech acts have also been adopted by research on software agents (Genesereth and Ketchpel 1994). This body of work depends fundamentally on agent communication languages (ACL), which are standardized sets of speech acts that ensure interoperability in agent to agent communication. The widely used FIPA ACL is based on speech act theory and has a logical semantics defined using multi-modal BDI logic. But there is no integration of speech acts within a more general framework of action and change that would facilitate planning of speech acts together with other actions to achieve goals. This is reflected by architectures that use the FIPA ACL-based JADE framework in robotic applications. Their focus is not planning but rather the interoperability that is gained from using a standardized ACL and the ease of developing agents using JADE.

In contrast, Morgenstern (1988) offers an integrated theory of both types of actions using a *syntactic* first-order logic that includes quotation. Davis and Morgenstern (2005) provide an alternative integration using regular first-order logic. The two theories' semantics cover both speech acts and their content, however their use has so far been limited to a STRIPS-like planner in the case of the former theory, and as a specification for future implementations in the case of the latter.

Temporal Action Logic

Reasoning about both physical and communicative actions is only possible in a highly expressive formalism. We have chosen one such formalism as a foundation, the Temporal Action Logic (TAL), and extended it with a representation of belief and commitment.

The origins of TAL are found in Sandewall's model-theoretic Features and Fluents framework (1994). Doherty (1994) selected important concepts, such as an explicit time line and the use of occlusion (discussed below), to form TAL and gave it a proof-theoretic first-order characterization. Doherty and Kvarnström (2007) provide a detailed account of the logic, but the version presented below includes further extensions that make TAL suitable for applications in multi-agent planning and reasoning.

TAL uses *fluents* to model properties and relations that may change over time. A fluent f is a function of time, and its value at a time point t is given by (value t f). An agent carrying out action a during time interval $(t_1\ t_2]$ is specified by a predicate (Occurs *agent* $(t_1\ t_2]$ a). But the most important feature of TAL is its *occlusion* concept. A fluent that is persistent by default is permitted to change its value when occluded, but must retain its value during time intervals when not occluded. The following formula (with free variables implicitly universally quantified and in prefix form to make the representation of the quoted formulas introduced below more convenient) relates a fluent f's value at the start and end time points of a time interval:

$$(\rightarrow (\neg\ (\text{Occlude}\ (t_1\ t_2]\ f))\ (=\ (\text{value}\ t_1\ f)\ (\text{value}\ t_2\ f)))$$

By assuming that fluents are not occluded unless otherwise specified, one is in effect making the frame assumption that things usually do not change. Exceptions are made explicit by occluding affected fluents in action specifications and dependency constraints. E.g., if the UAV flies between two locations, its location fluent (location uav) would be occluded during any interval with a non-empty intersection with the movement interval. By exercising fine-grained control over occlusion one gains a flexible tool for dealing with important aspects and generalizations of the frame problem.

Syntactic Belief

Previous accounts of TAL lack a representation of agents' mental states and beliefs. Introducing a *syntactic* belief operator that takes a *quoted* formula as one of its arguments provides a simple and intuitive notion of beliefs. E.g., the fact that the UAV believes, at noon, that there are five survivors in grid cell 2,3 of the area map can be expressed by the following formula (where 12:00 is syntactic sugar for a Unix time integer):

(Believes uav 12:00 '(= (value 12:00 (survivors (cell 2 3))) 5))

We use the quotation notation from KIF (Genesereth and Fikes 1992), which is a formal variant of Lisp's. An expression preceded by a quote is a regular first-order term that serves as a *name* of that expression. Alternatively one may use a back quote, in which case sub-expressions can be *unquoted* by preceding them with a comma. This facilitates *quantifying-in* by exposing chosen variables inside a back quoted expression for binding by quantifiers. E.g., we can use quantifying-in to say that there is some number that the UAV believes to be the number of survivors:

$$(\exists\ n\ (\text{Believes uav 12:00}\\ \text{‘}(=\ (\text{value 12:00 (survivors (cell 2 3))})\ \text{’},n))) \qquad (1)$$

Note that it is not the existentially quantified number itself, but the *name* of the number, that should occur as part of the quoted third argument of Believes. The quote preceding the comma ensures that whatever value n is bound to is quoted to produce that value's name.

However, if expressed as above, Formula 1 would be satisfied if the UAV believes all tautologies of the form $x = x$. To see this, simply replace n by the term (value 12:00 (survivors (cell 2 3))) and apply existential generalization. We need to add the requirement that the UAV's belief is not merely a tautology, but that it really *identifies* the number of survivors.

Moore (1980) suggests the use of *rigid designators* and Morgenstern (1987) modifies this suggestion slightly in her requirement that *standard identifiers* are known. We follow Morgenstern and use the syntactic predicate (Id x) to single out the name x as a standard identifier, adding the background knowledge that integers are standard identifiers. It is convenient to introduce a two argument predicate asserting that the second argument is a standard identifier for the first:

$$(\leftrightarrow\ (\text{Id}\ \text{‘’},x\ \text{‘’},y)\ (\wedge\ (=\ x\ y)\ (\text{Id}\ \text{‘’},y)))$$

Note the interaction between back quote and quote in ‘’,x and ‘’,y to make sure that the arguments of Id are *names* of the expressions. The initial back quote turns the following quote into the name of a quote, leaving the variables x and y free for binding. The resulting expression denotes the quoted version of whatever the variables are bound to rather than quoted variables that can not be bound at all. The use of quoted expressions as arguments to the Id predicate prevents substitution of identicals, as is required by opaque belief contexts.

If the UAV can *identify* the number of survivors we say:

(∃ *n* (Believes uav 12:00
‘(Id ’(value 12:00 (survivors (cell 2 3))) ”,*n*)))

Here, the occurrence of the Id predicate is nested in a belief predicate, necessitating *double* quotes on the variable *n*.

Speech Acts

Speech acts can be used to communicate beliefs to, and to incur commitment in, other agents. The extensions introduced above make it possible to reformulate Allen's speech acts (1988) in TAL using syntactic belief and commitment predicates. The type of information we will be interested in is beliefs about what a particular value is. This is straightforwardly communicated by standard identifiers. E.g., if the UAV wishes to inform the CCC that it is in the map's grid cell 2,3 at noon, it may plan an action of the following form:

(inform ccc ’(Id ’(value 12:00 (location uav)) ’(cell 2 3))) (2)

However, this is complicated when the CCC wishes to *ask* the UAV what its location is. Hintikka (1978), and many others, suggests viewing questions as requests for information. The CCC should thus request that the UAV perform the inform action in Formula 2. But since the CCC does not know where the UAV is, which is presumably the reason why it is asking, it can not know what action to request.

Again we follow Allen's directions and introduce an informRef action designed to facilitate questions of this type. The informRef action does not mention the value that is unknown to the CCC, which instead performs the following request:

(request uav ’(Occurs uav (*b e*]
(informRef ccc (value 12:00 (location uav)))))

The informRef preconditions require that the informing agent holds a belief about what the value that is being informed about is and the effects assert the existence of a standard name that the hearer believes the value has. Note that an agent that commits to *executing* the action schedules an *inform* procedure call, plugging in the sought value. In contrast, an agent that only *reasons* about the effects of the informRef action, as in the question example above, knows that it *will* come to hold a belief about the value, but need not yet have such a belief.

Automated Natural Deduction

The theory presented so far needs to be complemented with an automated reasoner. Earlier work with TAL made deductive planning possible through a compilation of TAL formulas into Prolog programs (Magnusson 2007). But Prolog's limited expressivity makes it inadequate for our present purposes. Instead, our current work utilizes a theorem prover based on *natural deduction*, inspired by similar systems by Rips (1994) and Pollock (1999).

Natural deduction is an interesting alternative to the widely used resolution method. A natural deduction prover works with the formulas of an agent's knowledge base in their "natural form" directly, rather than first compiling them into clause form. The set of proof rules is extensible and easily accommodates special purpose rules that make reasoning more efficient. E.g., we incorporate specialized inference rules for reasoning with quoted expressions and beliefs. This works well, though the quoted expressions in our examples are simple atomic formulas. Whether the approach will scale to an effective method of reasoning with more general uses of quotation is an open question.

Rules are divided into *forward* and *backward* rules. Forward rules are triggered whenever possible and are designed to converge on a stable set of conclusions so as not to continue generating new inferences forever. Backward rules, in contrast, are used to search backwards from the current proof goal and thus exhibits goal direction. Combined, the result is a bi-directional search for proofs.

Nonmonotonic reasoning and planning is made possible through an assumption-based argumentation system. The set of *abducibles* consists of negated occlusion, action occurrences, temporal constraints, and positive or negative holds formulas, depending on the current reasoning task. These are allowed to be assumed rather than proven, as long as they are not counter-explained or inconsistent.

For some restrictions on the input theory we are able to guarantee completeness of the nonmonotonic reasoning (Magnusson, Kvarnström, and Doherty 2009). But in the general case, when one cannot guarantee completeness of the consistency checking, we might conceivably fail to discover that one of the assumptions is unreasonable. However, this would still not be a cause of *unsoundness*, since we are using the sound system of natural deduction that keeps track of all assumptions and the formulas that follow from them. But it might result in plans and conclusions that rest on impossible assumptions. A conclusion Φ depending on an inconsistent assumption would in effect have the logical form $\bot \rightarrow \Phi$, and thus be tautological and void. This is to be expected though, due to the uncomputability of consistency checking. The most one can hope for is for the agent to continually evaluate the consistency of its assumptions, improving the chances of them being correct over time, while regarding conclusions as tentative (Pollock 1995).

Agent Architecture

Solving the scenario in the introduction requires a system with a tight coupling between planning, execution, and monitoring. Our architecture achieves this through logic-based planning that results in abductive frame assumptions (about the persistence of certain parts of the world) that are monitored during plan execution, as described below.

Planning

Planning is the result of proving a goal while abductively assuming action occurrences that satisfy three kinds of preconditions. The action must be physically *executable* by an agent during some time interval, the agent must have a belief that *identifies* the action, and the agent must be *committed* to the action occurring, at the start of the time interval:

(→ (∧ (Executable *agent* (*b e*] *action*)
      (Believes *agent b* ‘(ActionId ”,*action* ’,*actionid*))
      (Committed *agent b* ‘(Occurs ’,*agent* (’,*b* ’,*e*] ’,*action*)))
   (Occurs *agent* (*b e*] *action*))

Executability preconditions are different for each action and are therefore part of the specifications of an action.

The belief precondition is expressed by a single axiom:

(→ (∧ (Primitive *name*) (Id arg_1 id_1) ⋯ (Id arg_n id_n))
   (ActionId ‘(,*name* ,arg_1 ⋯ ,arg_n) ‘(,*name* ,id_1 ⋯ ,id_n)))

This captures Moore’s (1980) insight that knowing identifiers for the arguments of a primitive action is knowing that action. This condition prevents e.g. a stock market agent from planning to get rich by buying “the stock that will increase in value.” While theoretically correct, the plan is of no practical value unless the agent can identify some particular stock that will increase in value.

The time point at which an action is executed is also critically important. One would not want the agent to generate a plan to buy a particular stock “when it is lowest” and sell it “when it is highest.” Without additional information about when these events occur this plan is equally useless. However, it seems overly restrictive to require that the agent holds beliefs that *identify* the action occurrence time points. Actions that do not depend on external circumstances can be executed whenever the agent so chooses, without deciding upon an identifiable clock time in advance. Actions that do depend on external circumstances can also be successfully executed as long as the agent is sure to know the correct time point when it comes to pass. This is precisely what the concept of *dynamic controllability* captures. Following Vidal and Fargier (1999) we denote time points controlled by the agent by *b* and time points over which the agent has no control by *e*. The temporal dependencies between actions form a simple temporal network with uncertainty (STNU) that can be checked for dynamic controllability to ensure an executable plan.

Finally, the commitment precondition can be satisfied in one of two ways. Either the agent adds the action to its own planned execution schedule (described below), or it uses the request speech act to delegate the action to another agent, thereby ensuring commitment.

Execution

Scheduled actions are tied to the STNU through the explicit time points in TAL’s Occurs predicate. An STNU *execution* algorithm propagates time windows during which these time points need to occur (Morris and Muscettola 2000). Time points that are *live* and *enabled* with respect to the time windows are executed, i.e. they are bound to the current clock time and action occurrences scheduled at those time points are proved *dispatched* using the following axiom:

(→ (∧ (ActionId ‘’,*action* ‘’,*id*)
      (ProcedureCall self (*b e*] *id*))
   (Dispatch self (*b e*] *action*))

The ProcedureCall predicate is the link between the automated reasoner and the execution sub-system in that the predicate is proved by looking up the procedure associated with the given action and calling it. Note that it is the action’s identifier that is used in the procedure call. This guarantees that only integers, strings, and other standard identifiers are passed as arguments to procedures, and is necessary since the action’s arguments might depend on information gathering actions whose results were not available at the time of planning. Invoking automated reasoning on the above axiom, rather than simply performing the procedure call, allows the full power of theorem proving to be applied in finding standard identifiers for the procedure call arguments. This could be necessary in order to apply background knowledge to convert the arguments into the standard format, or if the arguments are only implicitly represented as a deductive consequence of explicit knowledge.

Monitoring

Executing the plan will satisfy the goal as long as fluent persistence assumptions hold up. But the real world is an unpredictable place and unexpected events are sure to conspire to interfere with any non-trivial plan. To detect problems early we continually evaluate all assumptions that are possible to monitor.

When a persistence assumption (in the form of a non-occlusion formula) fails it produces an occlusion percept that is added to the agent’s knowledge base. A truth maintenance system removes assumptions that are contradicted by observations and unchecks goals that were previously checked off as completed but that include a failed assumptions among their dependencies. This immediately gives rise to a plan revision and failure recovery process as the theorem prover tries to reestablish those goals.

If the proof of the unchecked goals succeeds, the revision will have had minimal effect on the original plan. A failed proof means that the current sub-goal is not viable in the context of the execution failure, and the revision is extended by dropping the sub-goals one at a time. This process continues until a revision has been found, or the main goal is dropped and the mission fails.

UASTech Delegation Framework

Procedure calls are carried out by the UASTech Delegation Framework (Doherty and Meyer 2007). But the actions are often still too high-level to be passed directly to the low-level system. An example is the action of scanning a grid cell using the infrared camera. This involves using a scan pattern generator, flying the generated trajectory, and applying the image processing service to identify humans in the video footage (as described in (Doherty and Rudol 2007)). The assumption is that, while not a primitive action in the low-level system, the scanning of a grid cell will always proceed in the manner just described so there is no need to plan its sub-actions. Such macro-actions are coordinated by specialized modules, in this case the scan coordinator.

The Delegation Framework implementation uses the Java agent development framework (JADE) and encapsulates the agent so that all communication is channeled through a standardized interface as FIPA ACL speech acts. Human operators, like those in the CCC, communicate through an interface like any other agent but use a graphical user interface that displays a map subdivided into grid cells through which they can ask questions, position no-fly zones or other constraints, and delegate requests to other agents. The resulting multi-agent system can consist of widely differing agents that are able to interact through standardized speech acts to help each other achieve complex goals.

Scenario Solution

We have implemented this theorem proving based agent architecture and applied it to the scenario described in the introduction. If the UAV's knowledge base was initialized at 12:00 and the CCC requests having information of the survivor count in map grid cell 2,3 at 13:00 the UAV produces the following plan (in addition to an STNU that relates qualitative time points):

(Schedule uav (b_1 e_1] (fly (cell 2 3)))
(Schedule uav (b_2 e_2] (scan (cell 2 3)))
(Schedule uav (b_3 e_3]
  (informRef ccc '(value 13:00 (survivors (cell 2 3)))))

The success of the plan depends on two abductive assumptions that were made during planning and that can be monitored during execution:

($\neg$ (Occlude (12:00 b_3] (radio uav ccc)))
($\neg$ (Occlude (e_1 b_2] (location uav)))

There is also an assumption of the persistence of the survivor count, though this is impossible for our UAV to monitor since it can not see the relevant area all at once. If one of the survivors runs off, then the plan will be modified to take the resulting body count discrepancy into account when it is discovered.

Suppose however that the large distance and mountainous terrain causes a radio communication break down while the UAV is scanning the area. The UAV perceives that the fluent (radio uav ccc) *was* occluded and the truth maintenance system successively removes incompatible assumptions and sub-goals until a revised plan suffix is found:

(Schedule uav (b_4 e_4]
  (informRef mob '(value 13:00 (survivors (cell 2 3)))))
(Schedule uav (b_5 e_5]
  (request mob
    '(Occurs mob (b_6 e_6]
      (informRef ccc '(value 13:00 (survivors (cell 2 3)))))))

The new plan involves requesting help from another mobile agent (mob). By communicating the survivor count to this "middle man," and requesting it to pass on the information to the CCC, the UAV ensures that the CCC gets the requested information.

Another set of assumptions now require monitoring:

($\neg$ (Occlude (oc 12:00 b_6) (radio mob ccc)))
($\neg$ (Occlude (oc 12:00 b_4) (radio uav mob)))

While the UAV is incapable of monitoring the other agent's radio communication, it will be monitored if that agent is also running our agent architecture. Unless further failures ensue, this concludes the successful completion of the given knowledge gathering assignment.

Conclusions

We have described a scenario that involves planning communication between agents, plan execution with monitoring, and plan revision to recover from an unexpected communication failure. Our solution uses speech acts formalized in an extension of Temporal Action Logic that includes syntactic belief and commitment operators, which are made possible through the use of a quotation mechanism. Plan generation and revision is carried out using an automated natural deduction theorem prover. The subsequent execution uses an STNU execution algorithm to dispatch actions in accordance with the plan's temporal constraints. Finally, an action dispatch mechanism links the automated reasoning system and a delegation framework that coordinates the execution of primitive actions on the physical robot platform.

Our framework makes extensive use of logic to meet the challenges of dynamic environments. While the use of logic as a *theoretical* foundation is relatively commonplace, we have constructed a *practical* logical agent architecture that uses theorem proving technology to plan and execute actions in a multi-agent setting.

Much work remains before the technology is sufficiently efficient and robust for larger scale applications. But there is great potential for using logical agents in both real and simulated worlds. This paper has explored a robotic search and rescue scenario. Another paper uses the same technology in intelligent computer game characters (Magnusson and Doherty 2008). We believe these and similar opportunities make continued effort worthwhile.

References

Allen, J. 1988. *Natural Language Understanding*. Redwood City, CA, USA: Benjamin-Cummings Publishing Co., Inc.

Davis, E., and Morgenstern, L. 2005. A first-order theory of communication and multi-agent plans. *Journal of Logic and Computation* 15(5):701–749.

Doherty, P., and Kvarnström, J. 2007. Temporal action logics. In Lifschitz, V.; van Harmelen, F.; and Porter, B., eds., *Handbook of Knowledge Representation*. Elsevier.

Doherty, P., and Meyer, J.-J. C. 2007. Towards a delegation framework for aerial robotic mission scenarios. In *Cooperative Information Agents XI*, 5–26.

Doherty, P., and Rudol, P. 2007. A UAV search and rescue scenario with human body detection and geolocalization. In *Australian Conference on Artificial Intelligence*, volume 4830 of *Lecture Notes in Computer Science*, 1–13. Springer.

Doherty, P. 1994. Reasoning about action and change using occlusion. In *Proceedings of the 11th European Conference on Artificial Intelligence*, 401–405.

Foundation for Intelligent Physical Agents. 2002. FIPA communicative act library specification. `http://www.fipa.org/specs/fipa00037/`.

Genesereth, M. R., and Fikes, R. E. 1992. Knowledge interchange format, version 3.0 reference manual. Technical Report Logic-92-1, Computer Science Department, Stanford University.

Genesereth, M. R., and Ketchpel, S. P. 1994. Software agents. *Communications of the ACM* 37(7):48–53.

Hintikka, J. 1978. Answers to questions. In Hiz, H., ed., *Questions*. D. Reidel Publishing Company. 279–300.

Magnusson, M., and Doherty, P. 2008. Logical agents for language and action. In *Proceedings of the 4th Artificial Intelligence and Interactive Digital Entertainment Conference*.

Magnusson, M.; Kvarnström, J.; and Doherty, P. 2009. Abductive reasoning with filtered circumscription. In *Proceedings of the 8th Workshop on Nonmonotonic Reasoning, Action and Change NRAC 2009*. UTSePress. Forthcoming.

Magnusson, M. 2007. *Deductive Planning and Composite Actions in Temporal Action Logic*. Licentiate thesis, Linköping University. `http://www.martinmagnusson.com/publications/magnusson-2007-lic.pdf`.

Moore, R. 1980. Reasoning about knowledge and action. Technical Report 191, AI Center, SRI International, Menlo Park, CA.

Morgenstern, L. 1987. Knowledge preconditions for actions and plans. In *Proceedings of the 10th International Joint Conference on Artificial Intelligence*, 867–874.

Morgenstern, L. 1988. *Foundations of a logic of knowledge, action, and communication*. Ph.D. Dissertation, New York University, New York, NY, USA. Advisor: Ernest Davis.

Morris, P. H., and Muscettola, N. 2000. Execution of temporal plans with uncertainty. In *Proceedings of the 17th National Conference on Artificial Intelligence and 12th Conference on Innovative Applications of Artificial Intelligence*, 491–496.

Perrault, C. R.; Allen, J. F.; and Cohen, P. R. 1978. Speech acts as a basis for understanding dialogue coherence. In *Proceedings of the 1978 workshop on Theoretical issues in natural language processing*, 125–132.

Pollock, J. L. 1995. *Cognitive Carpentry: A Blueprint for how to Build a Person*. Cambridge, MA, USA: MIT Press.

Pollock, J. 1999. Natural deduction. Technical report, Department of Philosophy, University of Arizona. `http://www.sambabike.org/ftp/OSCAR-web-page/PAPERS/Natural-Deduction.pdf`.

Rips, L. J. 1994. *The psychology of proof: deductive reasoning in human thinking*. Cambridge, MA, USA: MIT Press.

Sandewall, E. 1994. *Features and Fluents: The Representation of Knowledge about Dynamical Systems*, volume 1. Oxford University Press.

Searle, J. R. 1969. *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

Shanahan, M. 2000. Reinventing Shakey. In *Logic-Based Artificial Intelligence*. Norwell, MA, USA: Kluwer Academic Publishers. 233–253.

Vidal, T., and Fargier, H. 1999. Handling contingency in temporal constraint networks: From consistency to controllabilities. *Journal of Experimental and Theoretical Artificial Intelligence* 11(1):23–45.

Revisiting Choice Points in Branching Time Temporal Structures: Ontological Foundations of a Formal Theory of Time Travel

Leora Morgenstern
Courant Institute of Mathematical Sciences, New York University, New York, NY 10012, United States of America

Abstract

Although the phenomenon of time travel is common in fiction, film, and video games, little work in AI has been done on integrating time travel into a formal theory of action. This paper considers how one might develop a formal ontology and model that support inferences about time travel.

This paper is an extended abstract. A more complete version of this paper will be available soon at http://www-formal.stanford.edu/leora/tt .

Introduction: Overview and Motivation

Although the phenomenon of time travel is common in fiction, film, and video games, little work in AI has been done on integrating time travel into a formal theory of action.

Why hasn't there been much work in this area? Perhaps because there is much about time travel that seems contrary to the spirit of formal AI. There is, first of all, the fact that in AI formal theories of action, time, whether linear or branching, goes only forward, and that time travel by definition requires moving backward in time. Second, time-travel seems to lend itself to various paradoxes, such as the grandfather (or autoinfanticide) paradox (Barjavel 1943; Horwich 1987). Third, AI, even formal AI, is ultimately motivated by practical problems [1] and time travel is in the realm of science fiction.

Why might work in this area be nevertheless interesting for AI? Time travel has become increasingly popular in popular culture, including TV (*Lost*), movies (*Back to the Future, Bill and Ted's Excellent Adventure*), and video games (Chrono Trigger). At the same time, the field of AI and entertainment has grown significantly. When time-travel video games are developed, it would be good to have some notion of what makes a time-travel narrative coherent, and it would be useful to have some formal theory underpinning this idea.

In: Benedikt Löwe (*ed.*), LSIR-2: Logic and the Simulation of Interaction and Reasoning, Pasadena CA, U.S.A., 12 July 2009. pp. 59–63.

[1]Even esoteric problems, such as variations on the Yale Shooting Problem (Hanks and McDermott 1987), or on the Missionaries and Cannibals Problem (McCarthy 1998) are studied in formal AI because they shed light on, respectively, temporal projection and elaboration tolerance.

Scope of paper and methodology: This initial work focuses on ontological and conceptual issues. We examine a very simple problem of "fixing" the past, and begin to develop a model that could support these inferences.

We make the following simplifications: First, we focus on traveling to the past rather than the future. Second, we restrict time travel to one agent at a time. Traveling groups of agents would significantly complicate the details of the ontological structure. Third, we do not allow bringing back objects from the future.

We focus on the following conceptual problems:

1. In standard temporal ontologies, time moves forward and not back. Is it possible to use these ontologies to represent moving back in time?
2. There are various representational difficulties that time travel presents. Can different sets of people exist in some chronicles that do not exist in others? Can several instantiations of a person (or an object) exist? How can this be represented in the ontology?
3. Time travel is subject to various paradoxes, such as the predestination and autoinfanticide paradoxes. How these can be avoided or explained in the developed ontology?

Axiomatization will be the next step in this project, but is not covered in this paper.

Temporal Ontology

Working Examples

We will refer in this discussion to two simple working examples:

Working Example 1 John is an athlete with Olympic aspirations. He crosses a room using a route that has an obstacle in it and breaks his foot. This results in his going to rehab instead of going to the Olympics. Some time later, John is given the opportunity to travel back in time. He now crosses the room using a route without an obstacle, does not break his leg, and goes on to win the Olympics.

Working Example 2 The beginnning of this example is the same as Working Example 1. Sometime subsequent to John's rehab, he marries and has a child, who learns from an early age about the accident that dashed her father's Olympic hopes. When she grows up, she travels back in time to just before the foot-breaking incident, and removes the obstacle from the route he takes. John winds up going to and winning the Olympics.

Choosing a temporal model for reasoning about time travel

AI temporal ontologies are of two flavors: linear time, as in the event calculus (Kowalski and Sergot 1986; Miller and Shanahan 1994) and branching time, as in the situation calculus (McCarthy and Hayes 1969; Reiter 2001). Because linear time does not facilitate reasoning about alternate possibilities, it seems inherently unsuitable as the underlying ontology for time travel. Branching time, with its built-in structure of different possible futures and different possible paths through the temporal tree, seems to have better potential. Still, we are faced with a basic conceptual problem: time goes forward, and not back. The idea is that you can get to any point that exists in the subtree rooted at the point you are now. But in time-travel you want to go back. That is, you wish to move to a different part of the subtree. This isn't allowed within standard branching time structures, so a different way must be found.

We use a combination of several tricks to achieve what we need.

First, we move away from the implicit assumption in time-travel accounts that there is a path in the time tree that "really" occurs; that this path is in some sense subsumed when time travel leads to a new path that "really" occurs. In our paradigm, there is no sense of a "real" path through the time tree. Rather, at any point in the time tree, certain sets of paths — more precisely, certain subtrees of the time tree — are accessible by certain embodiments of agents. Some of these subtrees intuitively correspond to how the world might be if time travel were allowed.

Second, we embrace the idea of different embodiments of agents. Intuitively, an agent changes as he goes through different experiences, most notably by gaining knowledge. In some sense, then, one can say that an agent Q is different at a later time than at an earlier time. We will talk about different embodiments of an agent. Specifically, we will talk about the embodiment of an agent $Q(S_b, S_a)$, where there is a path segment from S_a to S_b, where S_a precedes S_b; this represents an agent Q at S_a who in his mind has lived all the way up to S_b. He knows what happens on the path segment from S_a to S_b. Intuitively, $Q(S_b, S_a)$ corresponds to the agent Q who has time-travelled back from S_b to S_a.

Third, we introduce (possibly finite) sets of time-travel points and re-entry points for particular agents. These correspond to the points at which we can think of time travel occurring. They are constrained as follows: An agent is associated with a time-travel point and a re-entry point only if there is a path segment from the re-entry point to the time-travel point. (Let us call this path segment the *reversible path segment*.)

It is important to note that there is no action that takes an agent from a time-travel point to a re-entry point. Rather, an embodiment of an agent will be characterized by the pair (time-travel point, re-entry point), indicating that the agent in this embodiment will have (at least some) knowledge of what has happened in the reversible path segment. In the simple case where the agent embodiment is an embodiment of the agent who performed the first step of the reversible path segment, the embodied agent will have the knowledge and the "memories" of an agent who has traversed the entire reversible path segment.

In subtrees that are rooted at re-entry points, there will be paths in which alternate embodiments of agents have active roles. These are the paths which intuitively correspond to time travel.

In the same way that there are actions accessible to Q at S_a, there are actions available to $Q(S_b, S_a)$, the embodiment of the agent who has "lived through" the path segment from S_a to S_b. The set of actions accessible to an agent Q at a point S_a will usually overlap with the set of actions accessible to $Q(S_b, S_a)$. However, the set of actions will not necessarily be identical, since $Q(S_b, S_a)$ may face different constraints at S_a than Q.

Developing the model

To see in detail how this would work, let us consider more carefully Working Example 1, which is depicted in Figure 1. For this initial work, we use the temporal ontology similar

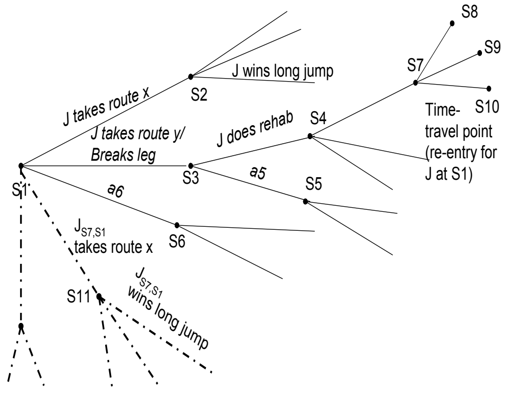

S8
S9
S7
J wins long jump
J takes route x
S2
S10
J takes route y/ Breaks leg
J does rehab
S4
Time-travel point (re-entry for J at S1)
S1
S3
a5
S5
a6
$J_{S7,S1}$ takes route x
S6
S11
$J_{S7,S1}$ wins long jump

Figure 1: *How time travel is effected in a classic forward-branching time tree. Certain time points are designated as time-travel or re-entry points for particular agents. Here S_7 is designated as a time-travel point, and S_1 is designated as a re-entry point for John. This means that an embodiment of John moves through a portion of the subtree rooted at S_1; the embodiment has full knowledge of what happened between S_1 and S_7*

to the situation calculus, as presented in (Reiter 2001). At time S_1, there are a variety of actions available to different agents.

Let us assume that there are several actions available to John at S_1, which include the action of crossing the room across route x (see the path segment from S_1 to S_2), and crossing the room across route y (path segment from S_1 to S_3). Route y contains an obstacle, and when John takes this route, he breaks his leg, dashing his dream of competing in the long jump in the Olympics. Instead, John must go to rehab. Poor John! Had he only taken route x, he would have gone on to win the long jump competition.

John continues in his dreary life, along the path whose initial segment is (S_1, S_3), always wondering what might have been. But then he reaches a time-travel point, labeled S_7 in Figure 1. What does time travel in this model actually constitute? As argued above, it would violate the principle of forward branching time to posit an action that John performs that takes him back to S and that would allow him to follow the time-tree path between S1 and S2, taking route x. Rather, we represent the time travel as an ordered triple (S_7, S_0, John). This triple represents the time-travel point, the re-entry point, and the agent who is intuitively doing the time travel, along portions of the subtree rooted at S_1. These portions of the subtree are accessible, not to the original John who had the choice of taking route x or route y, but to an older and wiser John, who knows the folly of taking route y. This is a *different embodiment* of John than the original John. This is the John who has known the pain of breaking a leg, going to rehab,and missing the Olympics. We denote this embodiment of John as $\text{John}(S_7, S_1)$, the John who has all the knowledge of what it is like to follow the path from S_1 to S_7, and chooses now to follow the better path.

$\text{John}(S_7, S_1)$ *cannot* follow the path from S_1 to S_2. That path was available only to John. But $\text{John}(S_7, S_1)$ can follow a very similar path, consisting of the same action, that of taking route x. This very similar path is depicted in Figure 3 as the path between S_1 and S_{11}.

Although we never identify any path through the tree as an actual path that happens, it is the case that in general an embodied agent $Q(S_b, S_a)$ will at S_a have the knowledge of someone who has traversed the path from S_a to S_b in the time tree.

We note the following about embodiments:

First, this process might be repeated along the path segment of an embodied agent, leading to nested embodiments. Consider what might happen after John wins the gold medal. He could make a bad choice and engage in illegal activities, such as the inhaling of illicit substances. This could lead to a fall from grace, and the cancellation of contracts for product endorsements. This is depicted in Figure 2, where the fall from grace occurs at S_{12}.

But this embodiment of John may be given yet another chance to redeem himself. Figure 2 shows that S_{14} is a time-travel point for $\text{John}(S_7, S_1)$ with a re-entry point of S_{12}. The embodiment of the twice time-travelling John is $\text{John}(S_7, S_1, S_{12}, S_{14})$; he is someone who behaves as a person who has traversed the time tree path segment from S_1 to S_7, has then traversed another time-tree path segment from S_1 to S_{14}, and begins again at S_{12}.

Second, as this example shows, embodiment along a path stays the same or grows deeper, but does not grow shallower. An agent is marked by the path segments that he believes he has lived.

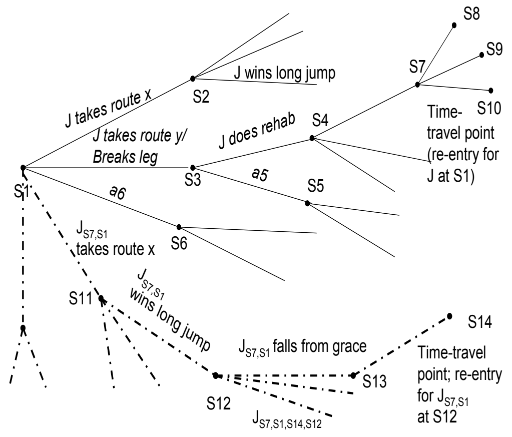


Figure 2: *Agent embodiments can be nested. In this case, there is a time-travel point along the path taken by the agent embodiment* $\text{John}(S_7, S_1)$. *This results in the nested embodiment* $\text{John}(S_7, S_1, S_{14}, S_{12})$

More on Agent Embodiments

We briefly discuss, though we do not necessarily resolve, the following questions:

Do other agents know that $Q(S_b, S_a)$ is distinct from Q?

Probably not, unless they have been involved in closely related time-travel. This is not even supported by the current model: it would at the least require time-travel and re-entry points for pairs of agents and complicated indexing schemes of embodiment that take into account multiple agents. It should, however, be possible for an agent embodiment to tell another agent (or agent embodiment) about the particulars of his embodiment, or equivalently, his time travels.

Is an agent embodiment $Q(S_b, S_a)$ at S_a the age that Q was at S_b, or the age that Q was at S_a?

Neither answer is entirely satisfactory. If $Q(S_b, S_a)$ is at S_a the age that Q was at S_b, then, if S_b is sufficiently greater than S_a, it will be noticeable that the agent that all know as Q has suddenly aged. If $Q(S_b, S_a)$ is at S_a the age that Q is at S_a, then the embodiment/time travel is hidden to all. The only problematic issue is that $Q(S_b, S_a)$ knows quite well how old he is. Presumably, however, in his mind the time travel has rolled back the years, and he finds it natural to be the age that Q is at S_a.

This brings up a related question:

Is it possible to have two or more embodiments of a single agent co-existing during a path segment? That is, can one meet a future or past version of oneself?

This is not necessarily impossible in the current model. Even in those path segments in which $Q(S_b, S_a)$ is the active agent, Q could be present as a passive agent. On the other hand, it seems easy to disallow it by constraining all path segments so that only one embodiment of an agent appears, in a sense to be made precise later.

Disallowing multiple embodiments of an agent to co-exist would seem to resolve Horwich's autoinfanticide paradox (Horwich 1987). However, the paradox can be reformulated in a more general form, as discussed below.

Is it possible for an agent to travel back to a time prior to his birth? In terms of the current model, can there be an agent embodiment $Q(S_b, S_a)$ if Q was not alive at S_a?

This question is central to Working Example II. At S_1, John's daughter has not been born. If her agent embodiment cannot exist at S_1, she cannot prevent John's fall.

More generally, enforcing such a constraint would disallow a great amount of time-travel literature, from E. Nesbit's *The Story of the Amulet* to the movie *Back to the Future*. We would not want to disallow these.

Yet allowing embodiments of agents that have not yet been born is also problematic. If the agent embodiment has a re-entry point x years before its birth, does it remain active for all these x years? In *The Story of the Amulet* and *Back to the Future*, time-traveling agents do not spend long in the past: they zoom back into the future from which they came, either when they have finished their mission, or when they feel endangered. But zooming into the future would be meaningless in the current model.

However, one could approximate the effects of zooming back into the future, after a brief visit to the past, by doing the following:
(1) Allowing entry-points of agent embodiments before the birth of the agent.
(2) Specifying activity periods for each agent embodiment.
(3) Enforcing very brief activity periods for agent embodiments in the period prior to the birth of the agent.

Elements of the Model

Summarizing our discussion above, the basic constructs of our ontology are:

- A partial ordered set of situations S with a least element S_0 such that for all situations S_i other than S_0, S_0 precedes S_i. If S_i precedes S_j, the interval between S_i and S_j is denoted $[S_i, S_j]$.
- A set Q of agents and agent embodiments, recursively constructed as described below
- A set of (possibly parameterized) actions, are functions on situations. At any situation, a (possibly empty) subset of actions is feasible for any agent or agent embodiment.
- A set of time-travel points $T \subseteq S$ and a set of re-entry points $R \subseteq S$.
- A 3-ary time-travel relation $RR \subseteq T \times R \times Q$.
- A set of constraints on the set of situations, agents and agent embodiments, and feasible actions. Constraints can be used, for example, to ensure that two or more embodiments of an agent do not co-exist, or that agent embodiments have limited activity in time periods before they are properly born.

We forsee using constraints to classify certain sets of paths in the time tree as obeying certain principles of coherence. For example, all sets of paths in which there are never two or more embodiments of a single agent within any interval obey the unique-agent principle. These notions of coherence will be used to enable inferences.

Agent embodiments are constructed as follows: If $(t, r, q) \in RR$, then $q(r, t)$ is an agent embodiment. If q is already expressed as an agent embodiment of the form $q'(r_1, t_1, ..., r_n, t_n)$, the newly constructed agent embodiment is expressed as $q'(r_1, t_1, ..., r_n, t_n, r, t)$.

We have not yet discussed the semantics for knowledge and belief. Since, effectively, most of the "time travel" is effected through the beliefs of the embodied agent, it is important to give a coherent semantics for this. We aim to give a modification of a standard possible-worlds semantics for belief. Instead of talking about accessible possible worlds or situations, however, it would seem preferable to talk about accessible time-travel structures, where a time-travel structure for an agent embodiment $Q(S_b, S_a)$ at S_c would include the path segment from S_a to S_b, as well as the path from S_a to S_c. Making this notion precise is ongoing work.

Time Travel Paradoxes and Puzzles

Time travel is subject to a number of well-known paradoxes and puzzles. Two of the best known are the autoinfanticide paradox (Horwich 1987; Barjavel 1943), and the predestination paradox.

The autoinfanticide paradox can occur in time-travel scenarios if one travels back in time and kills oneself when one is an infant. This would seem to entail that one would never have grown up and been able to time-travel back to the point when one is an infant: this seems paradoxical.

As discussed above it is easy to avoid the autoinfanticide paradox by positing that it is never the case that two or more embodiments of an agent co-exist in the same interval. However, this does not prevent the similar grandfather paradox, in which an embodiment of an agent kills the agent's grandfather before the grandfather gives birth to the agent's parent, thus ensuring that the agent will not be born.

The predestination paradox appears as far back as *Oedipus Rex*, though in that case without time travel: An agent, trying to avoid a prophesied and undesirable event, performs actions which in fact lead to the event happening.

It has been argued by philosophers and astrophysicists, such as Horwich and Novikov (Novikov 1998), that the autoinfanticide and grandfather paradoxes are not really paradoxes, since one can ensure that such branches do not occur. It appears to be feasible to formalize this intuition in the model that we are developing. We can specify sets of paths, such as ones in which an embodiment of an agent kills the agent's grandfather before the agent's parent is born, as incoherent. Or we can be sure to set up the action structure so that certain actions are not feasible for certain agent embodiments.

The predestination paradox is not really a paradox at all. Indeed, it may be desirable to set up one's time tree so that certain sets of paths are constrained to entail a particular fact (e.g., the occurrence of a particular event). Our model supports and encourage this phenomenon.

Related and Future Work

Time travel has long fascinated physicists and philosophers, who have thought about many of the issues discussed in this paper in great detail. Our contribution is to demonstrate how these notions can be integrated into existing AI theories of time and action.

To date, this is still preliminary work. Our next step is to formally categorize sets of paths or path segments that obey certain principles of coherence, and to demonstrate that desired inferences, such as the ones needed in Working Examples 1 and 2, hold in sets of coherent paths.

Future work includes lifting various restrictions and examining multi-person time travel and travel into the future. We would also like to explore to what extent results on regression (Reiter 1991) and progression (Lin and Reiter 1997) might still hold in this model. Although the temporal structure remains one that is strictly forward branching, there are many properties, such as the knowledge of the agent embodiments, that would seem to violate these results.

References

Barjavel, R. 1943. *Le Voyageuer Imprudent*. Gallimard.

Hanks, S., and McDermott, D. V. 1987. Nonmonotonic logic and temporal projection. *Artificial Intelligence* 33(3):379–412.

Horwich, P. 1987. *Asymmetries in Time*. MIT Press.

Kowalski, R. A., and Sergot, M. J. 1986. A logic-based calculus of events. *New Generation Computing* 4(1):67–95.

Lin, F., and Reiter, R. 1997. How to progress a database. *Artificial Intelligence* 92:131–167.

McCarthy, J., and Hayes, P. J. 1969. Some philosophical problems from the standpoint of artificial intelligence. In Meltzer, B., and Michie, D., eds., *Machine Intelligence 4*. Edinburgh: Edinburgh University Press. 463–502.

McCarthy, J. 1998. Elaboration tolerance. In *Working Papers of the Fourth International Symposium on Logical Formalizations of Commonsense Reasoning, Commonsense-1998*.

Miller, R., and Shanahan, M. 1994. Narratives in the situation calculus. *J. Log. Comput.* 4(5):513–530.

Novikov, I. 1998. *The River of Time*. Cambridge University Press.

Reiter, R. 1991. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In Lifschitz, V., ed., *Artificial Intelligence and Mathematical Theory of Computation: Papers in Honor of John McCarthy*. Academic Press. 359–380.

Reiter, R. 2001. *Knowledge in Action*. Cambridge, Massachusetts: MIT Press.