Active Logic and Practice

Malec, Jacek

Published in:
Linköping Electronic Conference Proceedings

2009

Active logic and practice

Jacek Malec
Department of Computer Science, Lund University
Box 118, 221 00 LUND, Sweden
jacek.malec@cs.lth.se

May 9, 2009

Abstract

The problem of finding a suitable formal approach to describe on-going reasoning process has been open since the very beginning of AI. In this paper we argue that active logic might be a formalism useful in this context. Active logic is first introduced, then we analyse resource limitations that constrain the space of possible practical realisations of such reasoners. Finally some steps towards creating a practical active logic reasoner are presented.

1 Introduction

The problem of finding a suitable formal approach to describe on-going reasoning process has been open since the very beginning of AI. In particular, the areas of reasoning about action and change, belief revision, defeasible reasoning, interleaving planning and acting in dynamic domains, have all addressed this problem, albeit partially, from different points of view. However, there is no well-developed theory of reasoning viewed as an activity performed in-time.

Another aspect of practical reasoning is that it is *always* performed with limited resources. Our approximations normally neglect this aspect, or address only some specific issue like real-time deadlines or limited memory footprint. But there is no formalism available yet that would provide a reliable starting point for building a reasoner able to tackle all the resource limitations occurring in practice.

One possibility, quite often adopted by practitioners, is to forget the theory and build systems that act irrespectively of the lack of appropriate formal grounds, suitable theory or complete explanation. There exist robots able to deal with apparently very complex dynamic environments (see e.g., results of the DARPA Urban Challenge [6]). The missing of theoretical grounds and incomplete formal reasoning, however apparent in many cases, do not preclude those systems from efficient, timely action. As a counterweight, an interesting attempt to base a practical system on well-founded grounds of formal reasoning, relevant in the context of this Workshop, is summarized in a recent thesis [14].

There have been numerous attempts to come up with theories of reasoning capable to be adapted to real-world constraints and limitations. We are not going to present them here but will focus on one particular approach worth reminding and, in our opinion, worth also further consideration. The paper will introduce active logic in the next section, then a short discussion of resource limitations will be presented. In the following section we will introduce our ongoing work in this area. Finally, some conclusions are stated.

2 Active Logic

The very first idea for our investigations [3] has been born from the naive hypothesis that in order to be able to use symbolic logical reasoning in a real-time system context it would be sufficient to limit the depth of reasoning to a given, predefined level. This way one would be able to guarantee predictability of a system using this particular approach to reasoning. Unfortunately, such a modification performed on a classical logical system yields a formalism with a heavily modified and, in principle, unknown semantics [18]. It would be necessary to relate it to the classical one in a thorough manner. This task seems very hard and it is unclear for us what techniques should be used to proceed along this line. But the very basic idea of "modified provability": *A formula is a theorem iff it is provable within* n *steps of reasoning*, is still appealing and will reappear in various disguises in our investigations.

The next observation made in the beginning of this work was that predictability (in the hard real-time sense) requires very tight control over the reasoning process. In the classical approach one specifies a number of axioms and a set of inference rules, and the entailed consequences are expected to "automagically" appear as results of an appropriate consequence relation. Unfortunately, this relation is very hard to compute and usually requires exponential algorithms. One possibility is to modify the consequence relation in such way that it becomes computable. However, the exact way of achieving that is far from obvious. We have investigated previous approaches and concluded that a reasonable technique for doing this would be to introduce a mechanism that would allow one to control the inference process. One such mechanism is available in Labeled Deductive Systems [12].

In its most simple, somewhat trivialized, setting a labeled deductive system (LDS) attaches a *label* to every well-formed formula and allows the inference rules to analyze and modify labels, or even trigger on specific conditions defined on the labels. E.g., instead of the classical Modus Ponens rule $\frac{A, A \to B}{B}$ a labeled deduction system would use $\frac{\alpha:A,\ \beta:A \to B}{\gamma:B}$, where α, β, γ belong to a well-defined language (or, even better, algebra defined over this language) of labels, and where γ would be an appropriate function of α and β. If we were to introduce our original idea of limited-depth inference, then γ could be, e.g., $\max(\alpha, \beta) + 1$ provided that α and β are smaller than some constant N.

A similar idea, although restricted to manipulation of labels which denote time points, has been introduced in *step-logic* [9] which later evolved into a family of *active logics* [11]. Such a restriction is actually a reasonable first step towards developing a formal system with provable computational properties. Active logics have been used so far to describe a variety of domains, like planning [17], epistemic reasoning [10], reasoning in the context of resource limitations [16] or modeling discourse. Quite recently there has been some successful work devoted to determining appropriate semantics for active logic systems [1]. However, only single-agent, static variant of the logic is covered there.

The real strength of active logic comes from the fact that labels are understood as discrete time points and that the set of premises used for reasoning may change with time. This way the formalism is prepared to accept fresh "observations" every "clock tick", thus extending the static logical consequence into the time dimension. Another very important aspect of active logic is its paraconsistency, together with some mechanisms allowing removal of contradicting formulae from the knowledge base. These two latter properties are crucial for modelling practical systems with resource limitations, see e.g. [15].

3 Practical Reasoner

Active logic implemented according to the original definitions referenced above unavoidably suffers from combinatorial explosion of the number of formulae associated with every time step. Locally, it is still a classical logical system. However, early in the work on active logic an idea of limited memory areas, somehow similar to human short-term memory, has been introduced [8]. The model is slightly more complex, with five memory banks fulfilling different functions (see Fig. 1). We have found it very appealing from the point of view of limited memory resources. In order to make it amenable for further analysis, we have formulated it as an LDS and tested its behaviour for hand, on some very simple examples [2, 4].

4 Paraconsistent Robot

In order to realise the idea of building a practical reasoner capable of taking account of time as it flows (in order to obey deadlines), capable of resolving inconsistencies in its knowledge base (due to e.g., erroneous observations corrected at some later point) and able to take into account its own limitations (memory size, processor speed or energy consumption) we have begun with a scenario involving

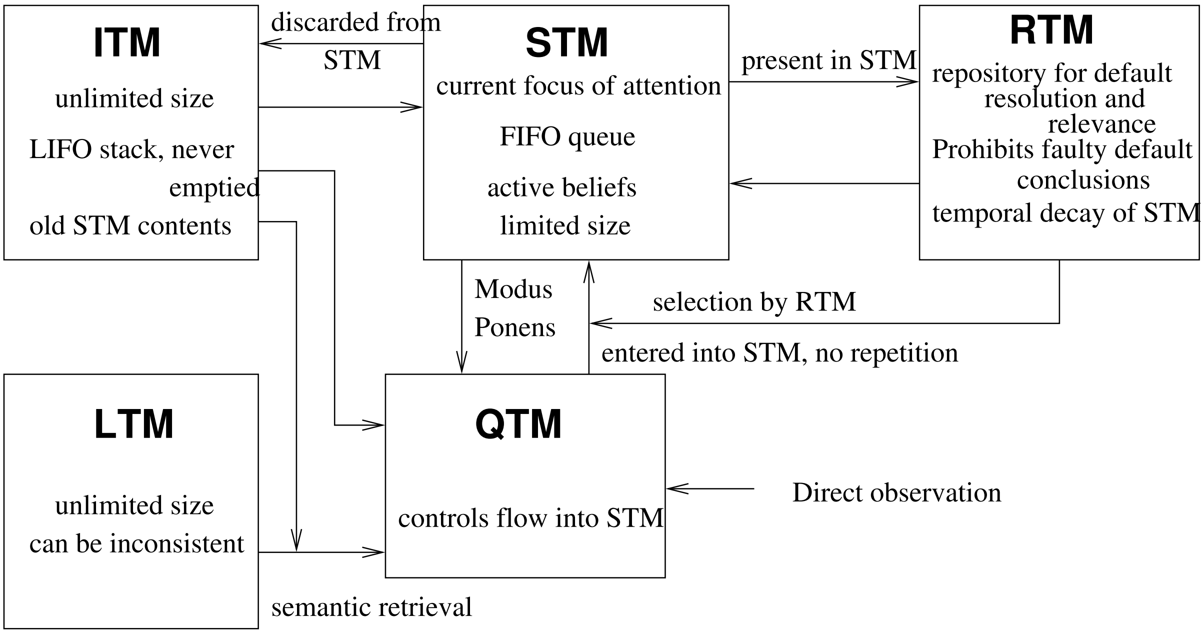

Figure 1: The memory model from [8].

a day of life of a service robot [13]. This scenario is loosely based on the much more interesting"Seven days in the life of a robotic agent" [5]. The idea is that a service robot has a series of tasks to be performed during the day, some of them more important than others and some provided with hard deadlines. A normal day plan for the robot allows the schedule (and all the deadlines) to be met. However, some day a problem occurs, requiring the robot to realise the problem and replan. The questions that might arise (in the rough order of complexity) are: What is the problem? Is the current plan inapplicable? Can I find a new plan to reach my goals? Can I find a plan meeting all the deadlines? Can I find one in time to meet deadlines (I can't reason too long then)? Can I find one given my current resources? e.t.c.

We have begun by creating a theorem prover capable to take an LDS specification (consistent with the formalisation from [2]). It has been shown to correctly prove a number of active logic theorems [7], in particular with observations coming as a stream of data while reasoning. Then we have applied it to our robot-day scenario [13], with the conclusion that in principle the prover is capable of performing the necessary reasoning, however it suffers from inefficient implementation and possible memory leaks. At this point we consider rewriting the prover again with speed as the major design objective.

5 Conclusions

In this paper we have briefly presented active logic and our work trying to apply it in scenarios relevant for continuous reasoning. In particular, active logic allows for reasoning in time, incorporating on the way an incoming stream of observations. It also lets us take care of inconsistencies. Embedding it in a mechanizable LDS allows us to take into account physical resource limitations, thus making the resulting system applicable in practice.

The original title of this paper was "Active Logic in Practice". However, I have realised that although this is my intention, it still requires a lot of both theoretical and practical effort to get to the point when we can say that active logic is usable in practice. In particular, we need to address the following points: efficiency of the prover; implementation on a robot, involving transforming its physical stream of sensory data into a symbolic stream of observations; further theory development: what do we really implement? how can this model be extended onto multiple cooperating agents? Those, and many other, questions require a lot of work before we can say that a practically useful reasoning system has been developed.

Acknowledgements

The author is indebted to Mikael Asker and Johan Hovold for their work on active logic theory, and to Sonia Fabre, Julien Delnaye and Thorben Ole Heins for their work on theorem provers.

References

[1] M. L. Anderson, W. Gomaa, J. Grant, and D. Perlis. Active logic semantics for a single agent in a static world. *Artificial Intelligence*, 172:1045–1063, 2008.

[2] M. Asker. Logical reasoning with temporal constraints. Master's thesis, Department of Computer Science, Lund University, August 2003. Available at `http://fileadmin.cs.lth.se/ai/xj/MikaelAsker/exjobb0820.ps`.

[3] M. Asker and J. Malec. Reasoning with limited resources: An LDS-based approach. In et al. B. Tessem, editor, *Proc. Eight Scandinavian Conference on Artificial Intelligence*, pages 13–24. IOS Press, 2003.

[4] M. Asker and J. Malec. Reasoning with limited resources: Active logics expressed as labelled deductive systems. *Bulletin of the Polish Academy of Sciences, Technical Sciences*, 53(1), 2005.

[5] W. Chong, M. O'Donovan-Anderson, Y. Okamoto, and D. Perlis. Seven days in the life of a robotic agent. Technical report, University of Maryland, 2002.

[6] DARPA. Urban Challenge. `http://www.darpa.mil/GRANDCHALLENGE/overview.asp`, 2007.

[7] J. Delnaye. Automatic theorem proving in active logics. Master's thesis, Department of Computer Science, Lund University, June 2008. Available at `http://fileadmin.cs.lth.se/ai/xj/JulienDelnaye/report.pdf`.

[8] J. Drapkin, M. Miller, and D. Perlis. A memory model for real-time commonsense reasoning. Technical Report TR-86-21, Department of Computer Science, University of Maryland, 1986.

[9] J. Elgot-Drapkin. *Step Logic: Reasoning Situated in Time.* PhD thesis, Department of Computer Science, University of Maryland, 1988.

[10] J. Elgot-Drapkin. Step-logic and the three-wise-men problem. In *Proc. AAAI*, pages 412–417, 1991.

[11] J. Elgot-Drapkin, S. Kraus, M. Miller, M. Nirkhe, and D. Perlis. Active logics: A unified formal approach to episodic reasoning. Technical report, Department of Computer Science, University of Maryland, 1999.

[12] D. Gabbay. *Labelled Deductive Systems, Vol. 1.* Oxford University Press, 1996.

[13] T. O. Heins. A case study of active logic. Master's thesis, Department of Computer Science, Lund University, January 2009. Available at `http://fileadmin.cs.lth.se/ai/xj/ThorbenHeins/report.pdf`.

[14] F. Heintz. *DyKnow. A Stream-Based Knowledge Processing Middleware Framework.* PhD thesis, Department of Computer Science, Linköping University, Sweden, 2009. Linköping Studies in Science and Technology, Dissertation No. 1240.

[15] C. Hewitt. Common sense for concurrency and strong paraconsistency using unstratified inference and reflection. Technical report, `http://carlhewitt.info`, 2008. Available at: `http://commonsense.carlhewitt.info`.

[16] M. Nirkhe, S. Kraus, and D. Perlis. Situated reasoning within tight deadlines and realistic space and computation bounds. In *Proc. Common Sense 93*, 1993.

[17] K. Purang, D. Purushothaman, D. Traum, C. Andersen, D. Traum, and D. Perlis. Practical reasoning and plan execution with active logic. In *Proceedings of the IJCAI'99 Workshop on Practical Reasoning and Rationality*, 1999.

[18] B. Selman and H. Kautz. Knowledge compilation and theory approximation. *JACM*, 43(2):193–224, 1996.