Safe learning agents? Logic-based agents!

Eduardo Alonso
Department of Computing, City University, London EC1V OHB
eduardo@soi.city.ac.uk

Abstract

Safe learning agents are agents whose learned behaviour can be predicted and analysed. Non-symbolic learning algorithms such as reinforcement learning rely on emergence and thus are not a good candidate to building safe AI systems. Our contention is that logic-based algorithms such as explanation-based learning and inductive logic programming should instead be used to design and implement safe, intelligent agents.

1. Introduction

Intelligence implies a certain degree of autonomy, that is, the ability to make independent decisions. This ability becomes crucial in dynamic domains in which designers cannot possibly foresee all situations their systems (agents henceforth) might encounter. This is especially true for multi-agent systems, where Nature is not the only source of uncertainty but the presence of other agents with different goals, beliefs, and intentions. It seems that for the agents to be autonomous they must be provided with the appropriate tools to adapt to and learn from their environment. Learning is closely linked to the notion of autonomy and should thus be a major focus of agent and multi-agent research (Alonso *et al*. 2001).

On the other hand, if the agents learn from themselves and act autonomously, then they might end up executing unsafe actions, actions that were not taken into account when the agents were originally designed. To put it crudely, we want autonomous agents but not anarchic ones.

Needless to say, the *safety problem* applies not only to the learning mechanisms that the agents implement but to the agents themselves. We believe that one way of finding solutions to the particular problem of safe learning agents is to consider the more general problem of safe agents.

2. Reactive Learning Agents

The intractability problems with traditional symbolic/logic approaches to building agents (Chapman 1987) gave way in the mid eighties to the idea that rational behaviour is linked to the environment an agent occupies ---intelligent behaviour is not disembodied. Intelligent behaviour *emerges* from the interaction of various simpler behaviours. Agents thus do not do symbolic reasoning at all. Their behavior is implemented as rules of the form situation → action which map perceptual input directly to actions.

There are obvious advantages to reactive approaches: simplicity, economy, and computational tractability. However, as many researchers have pointed out (e.g., (Wooldridge 2000)), there are some fundamental problems with purely reactive architectures.

- The first criticism has to do explicitly with learning. Reinforcement learning (Q-learning to be precise) is *the* learning mechanism for learning agents (see (Sen and Weiss 1999) for an overview on learning agents). In reinforcement learning agents adapt their decision process based exclusively on environmental feedback resulting from their choice of actions. As a consequence, agents do not learn relevant information of their environment such as relational structures, but just attribute-value pairs. Not surprisingly, generalisation is difficult to achieve[1]. Reactive agents must therefore start to learn from scratch every time they are situated in a new environment. It is difficult to see how this approach can lead to building intelligent systems of *general* competence, that is, Artificial Intelligence systems.

- The fundamental criticism is about emergency though. Ultimately, there is no principled methodology for building such agents: one must use a laborious process of experimentation, trial, and error to engineer an agent. Being successful now does not guarantee that we will succeed next time. That is not safe engineering. In particular, when using non-symbolic learning techniques such as reinforcement learning, engineers cannot either predict or analyse the behaviour of their agents. Agents learn how to execute a set of actions (that is, they acquire procedural knowledge), but not what happened or why they were successful or failed (declarative and explanatory knowledge). That is definitely not safe learning. We identify then a safe agent as an agent

[1] This drawback also applies to some logic-based learning mechanisms: Decision tree learning algorithms such as ID3 (Quinlan 1986) and its successor C4.5 (Quinlan 1993) are quite poor due their propositional nature.

whose behavior is liable to be predicted and analysed. The same applies to *safe learning agents*. Safe learning agents are agents whose learning is predictable and analysable.

Of course, most agent architectures are not purely reactive. Since Georgeff and Lansky's developed their Procedural Reasoning System (PRS (Georgeff and Lansky, 1986)) based on a BDI architecture, reaction and deliberation come hand in hand. Our contention is that a logic-based approach is needed to designing, implementing, and evaluating intelligent safe agents, not that reaction is intrinsically unsafe. Serious attempts have been made to combine reinforcement learning with logical/relational learning (e.g., Relational Reinforcement Learning, (Dzeroski et al. 2001)).

3. Logic-Based Learning Agents

Like some AI pioneers (e.g. (Nilsson 1995)), we believe that it is time to come back to Good-Old Fashioned Artificial Intelligence (GOFAI). From an operational point of view, distributed AI (aka Multi-Agent Systems) has proven to solve some of the tractability problems which traditional symbolic approaches suffered from[2]. By distributing the computational load among different specialised agents, we gain both in speed (parallelism) and quality of the solutions.

Admittedly, distributed learning makes things a little more complicated. If the rules are learned from different sources, the system should guarantee that the rules are no in conflict with each other. Current distributed learning methods employ one of two general approaches:

- Hypothesis combination: Each agent individually computes a local hypothesis based on the local training data which are then combined into a global hypothesis, either by a separate agent (*e.g*., Fayyad et al. 1993), or collaboratively by the local agents (*e.g*., Provost and Hennessy, 1996).

- Hypothesis update: One agent starts by learning a local hypothesis from the local training data, and then communicating it to another agent which updates its hypothesis based on its own local data (Domingos 1996).

Regardless of the learning technique used, the safety of these approaches is still under investigation,. However, we think that symbolic approaches are better suited than non-symbolic approaches in terms of safety. Symbolic/logic approaches provide a clear, well-defined, and natural representational language in terms of both syntax and semantics. Moreover, theorem proving as a metaphor for reasoning is subject to analysis. Coherence and completeness will determine how safe a system is.

What about learning? Again, we believe that knowledge (logic)-based learning mechanisms would give us the appropriate tools to building safe agents. We will focus on two logic-based learning mechanisms, namely, explanation-based learning and inductive logic programming.

3.1. Explanation-Based Learning

Explanation-based learning (EBL) (Mitchell *et al.* 1986) has been widely used in Artificial Intelligence to speed-up the performance of planners (e.g, in Prodigy (Carbonell *et al.* 1990)). Generally speaking, the agents are concerned in improving the efficiency of the problem solver rather than acquiring new knowledge. Obviously, problem solvers, when presented with the same problem repeatedly, should not solve it the same way and in the same amount of time. On the contrary, it seems sensible to use general knowledge to analyze, or explain, each problem solving instance in order to optimize future performance. This learning is not merely a way of making a program run faster, but a way of producing qualitative changes in the performance of a problem-solving system.

In short, EBL extracts general rules from single examples by generating an explanation why the system succeeded or failed, and generalizing it. This provides a deductive (rather than statistical) method to turn first-principles knowledge into useful, efficient, special-purpose expertise. The learned rules enable the planner to make the right choice if a similar situation arises during subsequent problem solving. As a deductive (learning) mechanism, EBL is open to analysis and is thus safe. If the domain theory is correct and complete, certain algorithms such as PROLOG-EBG (Kedar-Cabelli and McCarty 1987) produce *justified* general hypothesis.

3.2. Inductive Logic Programming

It is one thing to deduce from general rules more rules that should, in theory, lead to a successful plan; it is another to learn from examples (and some prior knowledge) whether the conditions for such plan to be executed successfully do

[2] It has been shown that distributing problem solving over several levels of abstraction reduces the time complexity of the planning process (Korf 1987; Montgomery and Durfee 1993). In the same way, in a hierarchical MAS, the learning module can be applied within each abstract problem space. Since the abstract problem spaces define more general and thus simpler problems and theories, agents learn more efficient and appropriate rules (Knoblock *et al*. 1991).

actually hold. In contrast to EBL methods, inductive logic programming (ILP (Muggleton and de Raedt 1994)) computes a hypothesis not just based on general rules known beforehand but also on external and initially unknown circumstances. Generally, relying exclusively on EBL-generated rules can turn out to be impractical in real-world domains in which agents work with incomplete knowledge, and thus ILP is an important addition to the system's effectiveness.

ILP methods compute an inductive hypothesis with the help of training data (positive and negative examples) and background knowledge. As depicted in Figure 1, agents collect training examples based on executed actions and plans and their outcome. This, together with the domain knowledge base and a target predicate (e.g., success or failure) forms the basis of the inductive learning process which computes a hypothesis (i.e., a definition of the target predicate) that can be used in subsequent planning.

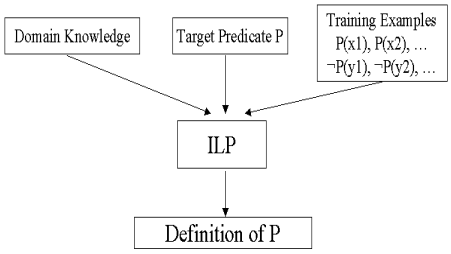

Figure 1. Inductive Logic Programming

Whereas traditional computational logic theory describes deductive inference from logic formulae provided by the user, ILP theory describes the inductive inference of logic programs from instances and background knowledge. In this manner, ILP may contribute to the practice of logic programming, by providing tools that assist to develop and verify programs (agents). ILP is based upon inductive inference rules that yield to a ''proof theory'', namely, inverse resolution (Muggleton 1995). As long as ILP is based on a strong theoretical background that allows analysis, we can conclude that it might lead to safe learning.

4. Conclusions and Further Work

In this paper, we have argued that safe agents are those agents over which the designer might exercise some control. Admittedly, in complex, dynamic domains, it is impossible to have complete control over the domain. However, designers should be able to analyze the behavior of their agents and to predict it more accurately as time passes. It is our contention that logic-based agents are the sort of agents that are liable to be analyzed in such way. Therefore, logic-based agents are *safe*r than non-symbolic, reactive agents. Following this line of argumentation, we believe that logic-based learning mechanisms are safer than non-symbolic learning techniques such as reinforcement learning or neural networks.

We are developing a conflict simulator to prove this hypothesis (Alonso and Kudenko, 1999; Kudenko and Alonso 2001). The idea is to show that EBL and ILP are safe learning mechanisms in highly risky domains, such as a simulated battlefield. Tara A. Estlin has also studied how to apply multi-strategy learning (EBL plus ILP) to improve planning efficiency and quality.

References

Alonso, E. and Kudenko, D. 1999. Machine learning techniques for adaptive logic-based multi-agent systems. In *Proc. UKMAS-99*.

Alonso, E., d'Inverno, M., Luck, M., Kudenko, D. and Noble, J. 2001. Learning in Agents and Multi-Agent Systems*, Knowledge Engineering Review*. Forthcoming.

Carbonell, J., Knoblock, C. and Minton, S. 1990. PRODIGY: An integrated architecture for planning and learning. In van Lehn, K. (ed.) *Architectures for Intelligence*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Chapman, D. 1987. Planning for conjunctive goals. *Artificial Intelligence*, 32(3):333-377.

Domingos, P. 1996. Efficient specific-to-general rule induction. In *Proc. Of the Second International Conference on Knowledge Discovery and Data Mining*.

Dzeroski, S., De Raedt, L. and Driessens, K. 2001. Relational Reinforcement Learning. *Machine Learning*, 43(1/2): 7-52.

Estlin, T. 1995. *Using Multi-Startegy Learning to Imporve Planning Efficiency and Quality*. PhD thesis, University of Texas at Austin.

Fayyad, U., Weir, N. and Djorgovski, S. 1993. SKICAT: A machine learning system for automated cataloging of large scale sky surveys. In *Proc. of the Tenth International Conference on Machine Learning*.

Georgeff, M.P. and Lansky, A.L. 1986. Procedural Knowledge. In *Proc. of the IEEE*, 74: 1383-1398.

Kaelbling, L.P., Littman, M.L. & Moore, A.W. 1996. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research* 4: 237-285.

Kedar-Cabelli, S. and McCarty, T. 1987. Explanation-based generalization as resolution theorem proving. In *Proc. of the Fourth International Workshop on Machine Learning*, 383-389. San Francisco, CA: Morgan Kaufmann.

Knoblock, C.A., Minton, S. and Etzioni, O. 1991. Integrating Abstraction and Explanation-Based Learning in PRODIGY. In Proceedings *of The Ninth National Conference on Artificial Intelligence AAAI-91*, 93-102. Menlo Park, CA: AAAI Press.

Korf, R.E. 1987. Planning as search: A qualitatuve approach. *Artificial Intelligence* 33(1): 65-88.

Kudenko, D. and Alonso, E. 2001. Machine learning techniques for logic-based multi-agent systems. *Proceedings of the First Goddard Workshop on Formal Approaches to Agent-Based Systems*, NASA Goddard Space Flight Center, MD, USA. Springer-Verlag.

Montgomery, T. A. and Durfee, E. H. 1993. Search reduction in hierarchical distributed problem solving. *Group Decision and Negotiation* 2: 301-317.

Muggleton, S. and de Raedt, L. 1994. Inductive logic programming: theory and methods. *Journal of Logic Programming* 19: 629-679.

Muggleton, S. 1995. Inverse entailment and progol. *New Generation Computing Journal*, 13: 245-286.

Mitchell, T.M., Keller, R. and Kedar-Cabelli, S. 1986. Explanation-based generalisation: A unifying view. *Machine Learning* 1: 4-80.

Nilsson, N. 1995. Eye on the prize. *AI Magazine*.

Provost, F. and Hennessy, D. 1996. Scaling up: distributed machine learning with cooperation. In *Proc. of the Thirteenth National Conference on Artificial Intelligence, AAAI-96*.

Quinlan, J.R. 1986. Induction of decision trees. *Machine Learning* 1(1): 81-106.

Quinlan, J.R. 1993. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Sen, S. and Weiss, G. 1999. Learning in Multiagent systems. In G. Weiss, editor, *Multiagent systems: A modern approach to distributed artificial intelligence*. Mass: The MIT Press.

Wooldridge, M. 2000. Computationally grounded theories of agency. In E. Durfee, editor, *Proceedings of the Fourth International Conference on Multi-Agent Systems (ICMAS 2000)*. IEEE Press.