Collaborating to Refine Knowledge

Ciara Byrne
Department of Computing Science,
King's College,
University of Aberdeen,
Scotland AB9 2UE
byrne@csd.abdn.ac.uk

Peter Edwards
Department of Computing Science,
King's College,
University of Aberdeen,
Scotland AB9 2UE
pedwards@csd.abdn.ac.uk

1 Abstract

A group of intelligent agents may work together in order to solve a problem or achieve a common goal. The group may fail to achieve a goal due to the actions of one or more agents. The agents should be able to adapt their behaviour to ensure that such a failure is not repeated. One way in which they can do this is by using machine learning techniques to refine their knowledge. Our research is concerned with how they can do this effectively.

What to Learn: Learning in an intelligent system should improve the performance of that system. The performance of a community of intelligent agents may be evaluated by its coherence or by the success of individual agents in achieving their goals. "Coherence will refer to how well the system behaves as a unit." [1]. Cohesion may be measured along several dimensions. Among these are the quality of the solutions which the system produces, the efficiency with which solutions are produced and how gracefully performance degrades in the presence of failure or uncertainty. The perspective from which performance is evaluated will determine what it is useful to learn. In some circumstances, agents may improve their performance by learning to successfully compete for resources. On the other hand, if the performance of a group of cooperating agents is evaluated by its efficiency, the aim of learning may be to improve the mechanisms used to coordinate the actions of agents. When deciding what to learn, important factors to consider are the behaviours which the agent should exhibit (in order to perform well) and the characteristics of the agent's environment. The ability to learn should be included in the design of an agent architecture as a means of producing desirable behaviours. For example, a useful behaviour may be the avoidance of conflicts with other agents and an agent may learn to predict when a conflict is likely to occur. Fig.1 shows a behavioural ecology triangle [2]. If the agent designer knows what behaviours are required and the characteristics of the agent's environment (two vertices are fixed) she can solve for an agent design. The most obvious type of knowledge to learn from another agent concerns that agent's characteristics, i.e. its capabilities, responsibilities, how it chooses goals and constructs plans, etc. An agent can use this kind of knowledge to coordinate its activities with others by accurately predicting their behaviour, avoiding conflicts, deciding what information to communicate to them, etc. The more an agent interacts with another agent, the more it is likely to benefit from learning about that agent.

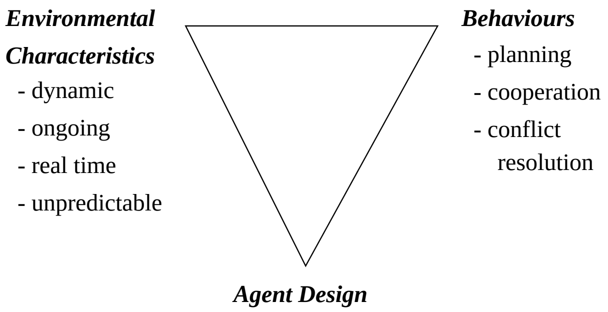

Figure 1: Behavioural Ecology Triangle

How to Learn: How an agent can learn from another will depend on several factors: the relationship between the agents, the knowledge representation used by agents and the type of knowledge to be learnt. If agents cannot communicate then one agent can learn from another only by observation. On the other hand if agents can communicate, they can share information directly. An agent may provide "ready-made" knowledge, e.g. a list of its own capabilities. It may also supply information which the other agent can combine with its own knowledge in order to learn, e.g. an example of a particular domain concept. Agents must use the same knowledge representation or be able to use some translation mechanism if the information exchanged is to be understandable to them. For example, KIF (Knowledge Interchange Format) [3] is an interlingua which has been proposed as a means of sharing knowledge between agents.

A Distributed Refinement System: The aim of our refinement system [4] is the production of coordinated and effective behaviour in an agent group. This can be achieved by resolving incompleteness and inconsistency in the agents' knowledge bases. We would hope that the performance of the agent group in achieving common goals would be improved as a result. Extensive investigation of techniques for refining the knowledge held in a single knowledge base has already been carried out [5] [6] [7]. The process of refining multiple related knowledge bases, such as those of a group of agents, presents new challenges. The refinement of one agent's knowledge may affect other agents in the system. The organisation of the agent community, the roles and expectations of individual agents, strategies used in cooperation, etc. amounts to a new body of knowledge that does not exist in a single-agent system, but which may also need to be refined. Agents are autonomous or semi-autonomous entities; their knowledge and reasoning processes are not necessarily transparent to other agents. Therefore, agents must volunteer information about their internal processing. We believe that a group of agents can more accurately determine the causes of a fault and implement an effective set of refinements if they cooperate by sharing their knowledge and different perspectives on a failure.

What is Learnt: Agents in our system are written in an Agent-Oriented Programming Language, AgentK* [8] which is based on Agent-0 [9]. The state of an agent consists of its current commitments, beliefs and capabilities. A belief is a statement which the agent considers to be currently true or false, e.g. "it is raining". Beliefs may change over time. An agent's capabilities are the actions which it can perform. The formation of a commitment by an agent obliges it to attempt to perform a particular action at a given time. Agents written in Agent-K* have the following basic types of knowledge:

- Capabilities: A list of the actions which the agent believes that other agents can perform.
- Commitment Rules: These are used to form appropriate commitments in response to messages from other agents.
- Goal Tree: Used to decompose the agent's overall goal into subgoals and eventually a sequence of primitive actions. When an agent wants to achieve a group goal, it requests the cooperation of appropriate agents. If they agree to participate, it sends them instructions during cooperation and dissolves the group either when the group goal has been achieved or cannot be achieved. All agents except the agent which initiated cooperation suspend their own goals during cooperation.
- Group Goals: Definitions of goals whose achievement requires the cooperation of several agents. Goals are described by the number and types of agents whose cooperation is needed and the actions which each of these agents should perform.
- Precondition Sets: An action has one or more precondition sets associated with it. At least one of these must be satisfied before the action will be performed.
- Domain Hierarchies: Knowledge, in the form of a hierarchy of Prolog predicates, about various aspects of the agent's environment or task domain which are used when checking precondition sets.

Agents learn refinements to their own knowledge. The form of a refinement will depend on the type of knowledge that is being refined. For example, refining knowledge about capabilities could involve adding or removing the belief that a particular agent can perform an action, whereas the definition of a group goal could be changed by reallocating an action to a different agent. To begin with, we are working on operators for generating refinements to precondition sets.

How Knowledge is Refined: Any agent written in Agent-K* can communicate with any other (provided it knows the agent's Universal Resource Locator). As all agents have the same architecture, and therefore use the same knowledge representation, they can exchange knowledge without the need for translation. To provide an agent with some information which it can use to generate refinements to precondition sets, Agent-K* allows agents to record information about the circumstances in which actions have been taken in the past. This information concerns the state of other agents as well as the state of the environment. Each record of an action includes the time it was performed, its arguments, who requested the action (the agent performing the action or another one) and a list of the agents with whom this agent was cooperating at the time. These records, together with other information about the state of the environment (e.g. current positions of objects), are used to generate appropriate refinements.

To allow agents to effectively refine their collective knowledge, we have introduced a new type of agent termed a refinement facilitator. A facilitator coordinates interaction between agents. For example, KQML [10] communication facilitators are used to manage message traffic among other agents by routing messages to appropriate agents, providing buffering and translation facilities, etc. There are several stages in the refinement process:

1. One of the agents participating in cooperation, usually the initiator, recognises that a failure has occurred. It does this by observing the other agents participating in cooperation and receiving information from them concerning their performance. It sends a description of the failure to the refinement facilitator.
2. The facilitator analyses the description of the failure and identifies the possible failure points.

3. The facilitator requests refinements from appropriate agents.

4. Agents propose refinements and send them to the facilitator.

5. The facilitator sorts the proposed refinements into sets. Refinements may be equivalent, conflicting or complementary. Two different refinements may have the same effect with respect to correcting a fault, but involve different changes to the knowledge of agents. Such refinements are considered to be equivalent. Refinements conflict if they cancel each other's effects and complement each other if both (or several) are needed to repair a fault. Each set of refinements consists of a number of complementary refinements.

6. The facilitator calculates ratings for refinement sets. It then chooses the highest rated refinement set and the appropriate refinements are implemented by agents.

7. Finally, the facilitator informs relevant agents about the refinements which have been made in order to maintain consistency between agents' knowledge bases (where this is necessary).

The use of a facilitator allows the views of several agents to be used in order to refine the collective knowledge of the agent group. In this system, there are several ways in which an agent could benefit from learning from another agent:

- Agents may suggest refinements to another agent's knowledge. For example, *Agent1* needs to refine its knowledge about the capabilities of *Agent2* so *Agent2* provides the appropriate knowledge.

- Agents may provide confidence ratings for refinements proposed by other agents. As an example, if *Agent1* failed to perform an agreed action during cooperation, it may propose a refinement to the precondition set associated with the action. *Agent2*, which requested the help of *Agent1* in order to achieve a goal, may be asked by the facilitator to calculate its confidence in this refinement. This confidence factor can be used by the facilitator when rating refinements.

- Agents could also exchange information in order to generate better refinements. For example, if *Agent1* has many records of the use of a particular action and *Agent2* does not, *Agent1* may share some of these records with *Agent2*. *Agent2* may use these records to generate better refinements to its own knowledge.

Future Work: It should be emphasised that this is very much work in progress and our ideas will develop as we put them into practice. To date, we have concluded that agents need information on the context in which they take actions in order to generate refinements to precondition sets. We have made a first attempt to facilitate the continuous generation of such information by extending an existing agent language. We have implemented a basic hunter-prey scenario by programming a group of agents in this language. We are currently integrating refinement operators into agents. Our aims in the immediate future include the development of a language for describing faults and refinements, and the construction of a prototype refinement facilitator.

References

[1] Alan H. Bond and Les Gasser, editors. *An Analysis of Problems and Research in DAI*, pages 3–35. Morgan Kaufmann, 1988.

[2] P.R. Cohen, M.L. Greenberg, D.M. Hart, and A.E. Howe. Trial by Fire: Requirements for Agents in Complex Environments. *AI Magazine*, pages 33–48, 10(3), 1989.

[3] R. Fikes, M. Cutkosky, T. Gruber, and J. V. Baalen. Knowledge Sharing Technology Project Overview. Technical Report KSL-91-71, Knowledge Systems Laboratory, Stanford University, 1991.

[4] C. Byrne and P. Edwards. Refinement in Agent Groups. In *Proceedings of the IJCAI-95 Workshop on Learning and Adaptation in Multiagent Systems*, 1995.

[5] D. Ourston and R.J. Mooney. Changing the Rules: A Comprehensive Approach to Theory Refinement. In *Proceedings of the Eighth International Conference on Machine Learning*, pages 485–489, 1991.

[6] B.L. Richards and R.J. Mooney. Learning Relations by Pathfinding. In *Proceedings of the Tenth National Conference on Artificial Intelligence*, pages 723–738, 1992.

[7] S. Craw and D. Sleeman. The Flexibility of Speculative Refinement. In L. A. Birnbaum and G. C. Collins, editors, *Machine Learning: Proceedings of the Eighth International Workshop*, pages 28–32, 1991.

[8] W. Davies and P. Edwards. Agent-K: An Integration of AOP and KQML. In Y. Labrou and T. Finin, editors, *CIKM Workshop on Intelligent Information Agents*. National Institute of Standards and Technology, Gaithersburg, Maryland, 1994.

[9] Y. Shoham. Agent-Oriented Programming. Technical Report STAN-CS-1335-90, Department of Computer Science, Stanford University, 1990.

[10] T. Finin, R. Fritzson, and D. McKay et al. An Overview of KQML: A Knowledge Query and Manipulation Language. Technical report, Department of Computer Science, University of Maryland, 1992.