Agent Models in Service of Information Dissemination in Inaccessible Cooperative Multiagent Systems

Olayide Olorunleke and Gordon McCalla

ARIES Lab, Department of Computer Science,
University of Saskatchewan, Canada.
{oto033, mccalla}@cs.usask.ca

Abstract. An agent is often required to act based on the information its environment or of other agents. It is possible that such information is not available to the agent even though another agent might have such information. Hence, a difficulty is knowing which of the other agents to contact for a piece of information that is required. We present a solution to this problem by using agent models that estimate the knowledge of other agents in conjunction with the predictive memory model of Bowling et al [1] to pro-actively send gossip to other agents. Empirical evidence observed from experiments suggest that a pro-active approach to information sharing can indeed be effective in making the relevant information available to the agents under certain assumptions.

1 Introduction

The state of the environment at any time t is defined to be a vector v_t, the contents of which may be accessed using the dot notation such as $v_t.x$, $v_t.y$, $v_t.z$, etc. With this information available, therefore, each agent can simply use its given policy for acting – which could have been learnt prior using reinforcement learning techniques – as a look-up table to select the optimal action in each perceived state. An environment is said to be *accessible* to an agent if the agent's sensors are always able to detect this current state – as long as the sensors have not failed. When this assumption of accessibility is broken, the environment is said to be *inaccessible*. Inaccessible environments by definition, therefore, make it difficult for the agent to select the optimal action even when it knows the optimal policy for acting in that environment.

One solution to this problem is to use a centralized model server that continuously receives information about the state of the environment (from the agents), aggregates these "snap-shots" into a more global picture and responds to each agent's query from this aggregated global perspective. An example is the PHelpS system [3]. This approach is often criticized, however, for providing a central point of failure for the system as a whole or for not scaling well with the number of agents [4].

Another solution is to use a decentralized approach in which each agent is responsible for managing its own models. In this case, each agent locally maintains information about the environment. The absence of the central model server implies that each agent takes over the responsibility of aggregating the information fragments that it needs. Among the problems introduced by choosing such a decentralized approach, the one we address in this paper is the following: *how does an agent locate another agent that has the relevant feature of the environment (or another agent) that it needs to make a decision?* In our attempt to answer this question, we make the assumption that the agents are in a cooperative multiagent system. With this assumption made then, we alternatively ask the question as: *how does an agent locate another agent that needs to know a feature of the environment (or a third agent) which this agent already knows?* Asking the latter question suggests a solution that relies on *pro-active informing* rather than the reactive one which the former question suggests. Reliance on gossip and observations alone, however, does not guarantee that the locally maintained models will contain enough up-to-date information to be used for choosing actions in inaccessible environments. For instance, an agent might receive no gossip and make no observations at all and will thus be left with incomplete information that cannot be used for action selection, or if used will result in an inappropriate selection for the current state of the environment. In such situations, Bowling et al [1] proposed the use of a mechanism called a *predictive memory*.

2 The Predictive Memory Model

The predictive memory model [1] maintains the state information – such as the position, velocity, direction, etc – of each object. Due to the possibility of not being able to always observe all of the state features, the predictive model also stores an additional value – for each state feature – that describes the accuracy of the current value assigned to it. In every time slice, the predictive memory approach updates the values of those features of objects that are directly observable from the sensory information obtained from the environment. Additionally, the probabilities attached to these values are set to 1.0. For those values that cannot be obtained from sensory input, the predictive memory approach uses two phases to achieve the update.

The first phase considers those changes that should occur based on the agent's own actions in the last time slice. For example, if the agent's last action was a turn by angle *a*, then it is possible to update the positions of the maintained data elements by correcting for this turn (since they are stored as relative positions).

The second phase in the update of the data stored in memory about the states features applies particularly to mobile objects. The assumption is that mobile objects tend to continue in their direction of motion and thus – even when out of view – there is a short-lived certainty that the objects will continue moving in that direction. Thus, the unseen mobile object's positions are updated using their last-observed velocities and positions. To account for this guess, the probability values attached to these unseen data items are reduced – by multiplying by a decay factor (e.g. 0.9). Additionally, the last observed velocity that was used to update the object's position is also decayed, to reflect the possibility that moving objects eventually slow down.

3 Enhancing the Predictive Memory via Gossip

To understand the processes involved in our use of gossip in this paper, consider the following definitions:

Definition 1:

if $v_t^A.x$ refers to the value assigned to feature x of the state vector at time t by agent A, and $C_{v_t^A.x}$ indicates how confident A is in this value, then we say that A is more confident than B about x at time t iff $C_{v_t^A.x} > C_{v_t^B.x}$.

Definition 2:

If agents A and B use the predictive memory approach, then they each have a subset (possibly empty) of features of the state for which they are more confident than the other. That is,

$$G_A^B = \{(x, v_t^A.x, C_{v_t^A.x}) \mid C_{v_t^A.x} > C_{v_t^B.x}\} \textit{ and } G_B^A = \{(x, v_t^B.x, C_{v_t^B.x}) \mid C_{v_t^B.x} > C_{v_t^A.x}\}$$

Definition 3:

When A gossips with B, the message $I_{A->B}$ that is sent by A to B must be a subset of G_A^B, i.e., $I_{A->B} \subseteq G_A^B$. If $|G_A^B| = 0$ then no message is sent.

Definition 4:

When B receives gossip from A about any feature x, it updates this feature in its predictive memory using the following:

$$C_{v_{t+k}^A.x} = decay^k * C_{v_t^A.x}; if\ C_{v_{t+k}^A.x} > C_{v_{t+k}^B.x}\ then\ v_{t+k}^B.x = v_t^A.x\ and\ C_{v_{t+k}^B.x} = C_{v_{t+k}^A.x}$$

Note that by this definition we assume that the message sent by A at time t will be received by B at $t+k$ (where it took k cycles for delivery).

Proposition 1: If A gossips with B about feature x, then B can do no worse if it accepts A's gossip into its predictive memory than if A did not gossip at all, and thus gossip can be used to enhance the predictive memory approach.

Proof. To prove this proposition, we consider two possible cases.

Case 1: When the feature value passed by A is accurate. In this case, B basically accepts accurate information or keeps its old beliefs. Either way, this is no worse than if A did not gossip at all.

Case 2: When the feature value passed is inaccurate. By definition, confidence values are only set to 1 when features are directly observed. If sensors are assumed to be perfect then a confidence value of 1 can be taken as a sign of accuracy. The case in which inaccuracy is evident then is when the confidence values are less than 1. Recall from definitions 2 and 3 that the contents of the messages are only drawn from those in which the gossiper's confidence is greater than the recipients'. This is only possible if the gossiper observed the feature at a certain time interval after the recipient had observed the same feature, thus making the gossiper's estimate more accurate. So the recipient (B) either changes to more accurate information or sticks with its original value. Thus, gossip can indeed be used to enhance the predictive memory approach. □

Despite the promise of gossip being useful in inaccessible environments, however, a few problems arise that must be tackled. The first of these arises from definition 2. That is, *A* is required to know both $C_{v_t^A.x}$ and $C_{v_t^B.x}$ in order to select G_A^B. Knowing $C_{v_t^A.x}$ is trivial since it is maintained locally by *A*. However, knowing $C_{v_t^B.x}$ presents a challenge for A because *B* maintains such knowledge locally. The second problem arises in definition 3. That is, how should the subset of G_A^B that is sent by *A* to *B* be determined?. The third problem arises from problems 1 and 2. That is, since *A* might not know *B*'s confidence in the value of feature *x* for certain, if *A* makes an estimate of this value and is wrong, then *x* would have made it into G_A^B even though *B* might actually be more confident than A about *x*. This in turn makes it possible that if *A* selects *x* to be in $I_{A->B}$ (i.e., *A* gossips with *B* about *x*), then by definition 4, *B* would reject *A*'s value for *x*. Although this error by *A* does not affect the predictive approach negatively, it results in a waste of communication bandwidth, and robs *B* of the opportunity to have received a value it actually needed.

To solve the first of these problems, we use agent models such that *A* keeps a model of *B* that only contains estimates of *B*'s confidence in each feature. Since we are unable to guarantee that the contents of the agent models are accurate, we propose to reduce the amount of wasted bandwidth by proposing the following heuristic.

Heuristic 1: *A* should also maintain a threshold for the modeled values such that it would only include *x* in G_A^B if $C_{v_t^A.x} > C_{v_t^B.x}$ and $C_{v_t^B.x} < threshold$. Note that in this case $C_{v_t^B.x}$ is obtained from *A*'s model of *B*.

4 Validation of the Gossip-Based Enhancements

Although we have formally proved that gossip can be useful in enhancing the predictive memory approach, we also verified this by running an experiment in the RoboCup domain in which a comparison is made between the time it takes to complete a task without proactive gossip and the time it takes with proactive gossip. The use of proactive gossip resulted in the task being completed in 70 cycles (down from 150 cycles without proactive gossip). We also wanted to confirm that the use of heuristic 1 could result in a reduction in the amount of bandwidth that is wasted by gossiping. We used a threshold value of 0.6 as described in heuristic 1. A total of only 15 extraneous messages were sent by the time the task was completed when heuristic 1 was used (down from 35 without heuristic 1).

5 Discussion and Conclusion

For this approach to managing and updating the locally held models to be transferable to other domains where models are distributed across agents, the assumptions we have made (which hold in the RoboCup domain) have to hold (or be made to hold where possible) in these other domains. The first of these is that the agents can observe each other. From this observation, it is possible to extract information that can be used to infer what an agent knows or does not know. For example, in an e-learning system, this could translate to allowing user agents to observe interactions that take place between each other. The second implicit assumption is the altruistic nature of the agents involved such that they are always willing to fill-in gaps in each other's knowledge when this is detected. This stresses the need for the research on persuasion and motivation already being pursued in [2], etc.; the results of which will help in ensuring that the agents or users are motivated to help each other. In this paper, agent models have been employed that contain estimates of what each agent knows. We have shown how these agent models can be used in decision making on when to gossip with other agents and in selecting the information that is included in the gossip. We have also shown a heuristic that can be used with these agent models to reduce that amount of bandwidth that is wasted when agents gossip with each other. We have not addressed the possibility that even though the agents do not lie about the information that is passed via gossip, such information can still be wrong (for instance, when sensors fail). In our current ongoing work we are studying various additional strategies [5] that can be used by the recipient to avoid believing the incorrect information it receives via gossip.

References

1. Bowling, M., Stone, P., and Veloso, M. (1996) Predictive Memory for an Inaccessible Environment. In Proceedings of the IROS-96 Workshop on RoboCup, pp. 28–34, Osaka, Japan, November 1996.
2. Cialdini R. (2001) Influence: The Science of Persuasion. Scientific American, pp. 76-81, February, 2001.
3. Collins, J., Greer, J., Kumar, V., McCalla, G., Meagher, P. and Tkach, R. (1997) Inspectable User Models for Just in Time Workplace Training. In A. Jameson, C. Paris, C. Tasso (eds.) User Modelling, *Proceedings of the 6th Conference on User Modelling, UM97,* Springer Wien, New York, pp. 327-337, 1997.
4. Fink, J. and Kobsa, A. (2000) A Review and Analysis of Commercial User Modelling Servers for Personalization on the World Wide Web. In User Modelling and User-Adapted Interaction, Special Issue on Deployed User Modelling, pp. 209-249, 2000.
5. Olorunleke, O. and McCalla, G. (2004) The Study of Delusion in Multiagent Systems. In Proceedings of Workshop on Modelling Other Agents from Observations (MOO 2004), pp. 33-40, New York, 2004.