Multi-Agent Coordination by Communication of Evaluations

Edwin de Jong
Artificial Intelligence Laboratory
Vrije Universiteit Brussel
Pleinlaan 2, B 1050 Brussels, Belgium
email: edwin@arti.vub.ac.be
tel: +32 2 629 37 00
fax: +32 2 629 37 29

Abstract. A framework for coordination in multi-agent systems is introduced. The main idea of our framework is that an agent with knowledge about the desired behavior in a certain domain will direct other, domain-independent agents by means of signals which reflect its evaluation of the coordination between its own actions and their actions. Mechanisms for coordination are required to enable construction of open multi-agent systems. The goal of this investigation was to test the feasibility of guiding an agent with coordination evaluation signals, and furthermore to gather experience with instantiating the framework on a testbed domain, the Pursuit Problem. In the testbed system, agents have been created which choose their actions by maximizing the coordination evaluation signals they will receive. The performance of these agents turned out to rank among the best results encountered in literature, and behavior guided by coordination evaluation signals can thus be concluded to be useful in this domain.

1 Introduction

Without a proper mechanism that guides interactions, the mere result of gathering many agents into a multi-agent system is chaos. Several such mechanisms exist. A short account of these is given in the next section.

Most of these mechanisms assume that the agents involved have direct access to the system in which they are incorporated. This implies that only agents that were designed specifically for the application at hand can be put to use. We want to investigate whether this restriction can be overcome. To this end, a coordination mechanism for multi-agent systems has been defined that allows domain-independent agents to behave usefully in an unknown environment. To learn how to do this, they are supplied with coordination evaluation signals that other, domain-specific agents send to them.

The framework defines how to model domains in a uniform way, by stating which classes have to be designed. In short, an Agent inhabits an Environment to which it is coupled by its Interaction object. Furthermore, it defines a way to coordinate an agent's behavior in an unknown environment by defining a coordination evaluation signal, which shall be sent by CoordinationSignalingAgents to CoordinationLearningAgents. The notion of domain independence throughout this paper refers to the notion of designing without knowledge of or making use of the domain in which the class, agents etc. in question, will be put to use. We do not claim our framework to be a general solution for coordination in any domain; the setup of teaching an agent to coordinate its actions by merely sending it evaluations of the appropriateness of their actions, restricts its scope for application to relatively simple domains with a limited number of possible actions.

If the approach is successful, it provides a way to control the large, open multi-agent systems that may be common in the near future. In the research that is reported on here, the goal was to test whether coordination evaluation signals are a viable approach to guide an agent's behavior. If this would turn out not to be the case, then the attempt to teach an agent to behave solely on the basis of these signals is doomed to fail. To test the feasibility of coordination based on coordination evaluation signals, we instantiate our framework on a testbed domain, and construct a perfectly rational agent that chooses its actions by maximizing the evaluation it will receive from other agents. The test is passed if this agent's behavior turns out to be appropriate in the testbed domain. In that case, our goal of having domain-independent agents learn to act usefully in an unknown environment by learning to maximize the evaluation signals they receive and relating them to the situation in that environment, appears to be an attainable idea.

The structure of the paper is as follows. First related work is discussed. Then the coordination signal framework is described. The description includes the object oriented model that is the basis for each application of the framework to an application domain. As an example, we demonstrate the instantiation of the framework to a domain, the pursuit problem as introduced by Benda [Benda et al., 1988]. Following that, we report on the experiments that have been conducted in this domain, and discuss the results. Finally, we point out the conclusions.

2 Related Work

In the literature, several mechanisms for coordination exist that can be utilized in multi-agent systems. Most coordination mechanisms for multi agent system rely on the exchange of structured information between agents. Whenever complex communication is used, the expectation that the recipient understands the messages implies that common knowledge is assumed. Here, we diminish such common knowledge by restricting communication to scalar values representing evaluations of an agent's behavior. In the literature, the ultimate extrapolation of this idea has been investigated by examing systems without any communication at all. With cooperation without communication as described in [Genesereth et al., 1986], choices of actions depend on knowledge of other agents' payoff functions.

In [Kraus and Rosenschein, 1992] and [M. Fenster and Rosenschein, 1995], coordination without communication is investigated using *focal points*, points to which the attention is drawn. The features that characterize these focal points (e.g. the first or last element in a row, or one element in a series that in some aspect differs from the others) are investigated. An important aspect of our framework which is not investigated in our paper, is learning. The idea that agents "learn which goals to pursue" was already mentioned in [Brazdil et al., 1991]. This idea has been investigated in [Sandip Sen and Hale, 1994] and furthermore in [Sen and Sekaran, 1995]. Sen e.a. use environmental feedback to enable agents to behave well [Sandip Sen and Hale, 1994]. Their approach is very much in line with ours, and their results of learning in a multi-agent system appear promising. The main difference is that their signals are environmental feedback, whereas ours are evaluations of coordination as perceived by other agents, and can therefore not make use of global knowledge of other agents' plans. Thus, an agent has to determine its evaluation factor locally. Nonetheless, apart from other agents' private knowledge (such as plans), agents are allowed to use global information.

Furthermore, *some* agents may send coordination and *some* agents may use them to choose their actions. These groups of agents may overlap.

In our setup, agents can try to influence other agents to act to their benefit. Also, it provides, at least in principle, the possibility to construct agents that act, guide other agents and are guided simultaneously.

3 The Coordination Signal Framework

In this section, the coordination signal framework is described. The description includes the steps that have to be taken to instantiate the framework in a domain. The main design goals of the framework were that it should be as general as possible, and that it should allow agents that have no knowledge of what behavior the system as a whole should show to choose their actions such that they play a role in achieving this behavior.

To this end, the capacity of the application-specific agents to evaluate the system will be used to teach the existing agents what action to perform in each situation. The object oriented design specifies two classes from which any agent in a system may inherit. These classes are the coordination-learning agent and the coordination-teaching agent. By having the application-specific agents communicate their evaluation of coordination with other agents to them, these other agents should in principle be able to relate this evaluation to the situation and their recent actions, and learn this relationship.

3.1 Coordination Evaluation Signals

The coordination signals that are exchanged between agents can be seen as a metaphor for the signals that humans beings use to express their thoughts about situations or people's actions. They can be seen as a simplification of the rudimentary vocabulary that people who speak different languages could use to coordinate their actions.

However, the coordination signals we use are scalar numbers, and bear no comparison to the subtle information that can be read from facial expressions, gestures or nods of approval. Still, the amount of information contained in such non-verbal communication is tiny compared to what can be expressed in a few lines of text. The analogy is made because the purpose of non-verbal communication is often the same; i.e. to express to other people what we think of their behavior, encouraging to continue or warning them for mistakes (see e.g. [Minsky, 1985], p. 280; "The function of laughing is to disrupt another person's reasoning!").

This restriction to a minimal language for agent communication may seem a voluntary restriction; people who speak the same language are obviously better equipped to cooperate. However, we judge such a modest vocabulary for agent communication to be more realistic than one which is based on the exchange of text. The reason is that exchanging text presupposes that the context and intellectual capacities of the recipient of the information are such that the message yields the effect that was anticipated by the sender because he *understands* this message. Since no existing machine can be seen to exhibit intelligence that is not restricted to quite limited domains, this assumption is clearly unrealistic.

As [Van de Velde, 1996] makes clear, an essential characteristic of a framework is that it places architectural commitments on instantiations that are supposed to comply to it. In this case, the main commitment is that agents are able to function without accessing domain-specific information *directly*. Naturally, in order to allow these agents to act usefully, they do need domain-specific information from the environment they inhabit, and they need a way to act. Both are provided by their Interaction object, which implements the interactions between agent and environment that are considered useful in a given environment, and couples these to the agent.

Another architectural commitment, is that the coordination evaluation factor may not depend on global knowledge of the agents' plans. This follows from the coordination evaluation's characteristic of being an *agent's* evaluation of coordination, as opposed to an environmental feedback.

3.2 Discussion of the Object Oriented Design

An object oriented design of the framework has been made. We will proceed to discuss the design, and explain how it can be used to instantiate the framework on a domain. Figure 1 shows the five central classes and follows the notation of [Rumbaugh et al., 1991]. A multi-agent system that complies with the the coordination signal framework is an instantiation of a subclass of the Environment class. It can contain agents, which all have their own Interaction object. The Interaction object represents an agent's interaction with the environment. It was designed to facilitate interactions between systems according to the paradigm of structural coupling. Structural coupling [Maturana and Varela, 1992] occurs, according to [Van de Velde, 1996] when two systems (agents in our case) "coordinate without exchange of representation, but by being mutually adapted to the influences that they experience through their common environment".

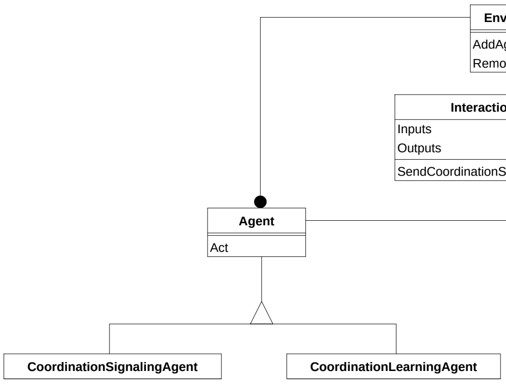

Fig. 1. OMT Object Diagram of Agents, the Environment and Interaction.

The underlying idea of the Interaction object, is that for internal processes of an agent, no essential difference exists between signals whose source is a perceptual process on one hand, and signals which will further on trigger actuators. The only characteristic of a signal that matters from the point of view of an internal process, is whether the process is the source or destination of the signal. Therefore, an interaction object has input and output signals.
Two classes are derived from the Agent class: CoordinationSignalingAgent and CoordinationLearningAgent. A CoordinationLearningAgent learns to coordinate its actions by relating the signals it receives from CoordinationSignalingAgents to the environment.

3.3 Instantiating the Framework to a Domain

To make an instantiation of the framework for a chosen domain, the following classes have to be designed.

- A subclass of Environment.
  This class contains all objects in the environment. The Environment class takes care of updating the interaction objects of the agents that are present. The method responsible for this can be overridden if necessary.
- A subclass of Interaction.
  This class should provide domain-specific methods to access and change the

environment. It also determines which objects and information from the environment are influencing the agent, and what effect the actions of the agents have. This is done by continuously feeding the inputs of the interaction object with signals from the environment and by reading the interaction object's outputs and interpreting them as actions, which may affect the environment. This mechanism allows agents to interact with unknown environments. Interaction objects furthermore store the evaluation signals that have been received.

- A subclass of CoordinationSignalingAgent.
  This will usually be a domain-specific agent. Agents of this class can evaluate the current situation in the environment, and as their name tells, they continuously send signals representing this evaluation to other agents, for example their neighbors.
- Optional: a subclass of CoordinationLearningAgent. Such subclasses should be reusable for different domains. An instantiation of CoordinationLearningAgent has to learn to interact with the environment using an Interaction object that was designed for that environment. The agent itself is not required to know anything about this environment, and just tries to figure out a relation between its interaction signals; this corresponds to how its perceptions are related and what effects their actions have on them. Thus, the agents of this class should be able to learn to choose the right actions in unknown environments. Furthermore, domain-specific agents can also be derived from this class. In that case, the framework is used to coordinate a group of similar agents in an environment they already know.

  Finally, the choice of the Coordination Evaluation Signals that will be sent by the CoordinationSignalingAgent subclass is an important factor. This signal is the only means for a CoordinationLearningAgent to learn to adapt its choice of actions to its environment. In general, it should reflect the degree to which an agent's actions are in line with those of its neighboring agents. The minimal complexity of the signal admittedly limits the scope of the framework to relatively simple problems. In our view however this is unavoidable for a general architecture at the current state of the art in AI. The possibility of increased insight into coordination in general compensates for this limited use in practical applications.

3.4 Implementation Issues

The domain-specific CoordinationSignalingAgents 'know' the environment in which they live. Conceptually, they interact with it via their Interaction object. To achieve this, we could include references to objects contained in a domain-specific Environment subclass in the Interaction subclass. For reasons of efficiency however, we allow them to directly access parts of the environment. This does not compromise the function of the Interaction class; its purpose is to provide a domain-independent way of interacting with environments, but since domain-specific agents have no need for this, we do not restrict their interactions with the environment to the interaction object.

4 Instantiation of the Coordination Signal Framework for the Pursuit Problem

In this section, we report on the instantiation of the coordination signal framework to a testbed domain, the pursuit problem. The pursuit problem is a well-known testbed problem from the Distributed Artificial Intelligence (DAI) literature. We have implemented this environment, and an agent that can evaluate the coordination between itself and its immediate neighbors. The agent, called MaxCoordinationPredator, simply compares for each possible action the coordination evaluations that would result from it, and choose the action that maximizes this evaluation. Thus, if this evaluation is a correct one, it is clear that this agent behaves rational.

4.1 The Pursuit Problem

We based our implementation on the description of the game and definitions of performance measures that are described in [Stephens and Merx, 1990]. We will first give a short description of the game.

- Start
  A rectangular, 4-connected grid of 30 x 30 contains a single prey in the middle. This prey chooses randomly between its possible actions: staying where it is, or moving in one of the four directions (or less, if some directions are blocked). Diagonal moves are not allowed. The predators are placed at random positions. At each time-step, the prey may move first. Then the predators move. They move one after another, thus avoiding collisions.
- Outcomes
  Possible outcomes are capture, stalemate and escape. A capture occurs when the four positions around the prey (left, right, above and below) are occupied. If the prey tries to move beyond the border, it's an escape unless two predators occupy two of these four positions, in which case it's a stalemate.
- Performance Measures
  Stephens and Merx use three performance measures: the Capture Ratio, Success Ratio, Success Efficiency.

Stephens and Merx base their strategies on captures positions. Capture positions are the four positions surrounding the prey. They examine three strategies. The one with least communication is *local control*, where an agent notifies the other agents when it occupies a capture position. The second strategy is *distributed control*. There, intentions (the capture position an agent wants to occupy) are transmitted before the move cycle. Finally, with *central control*, one agent commands the other agents.

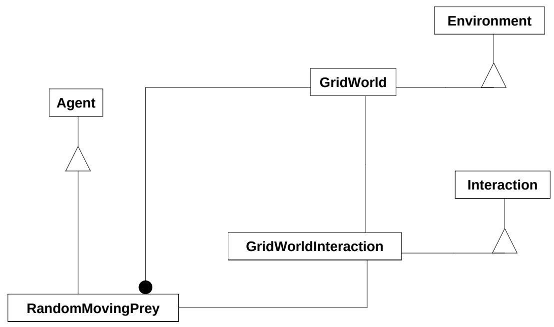

Fig. 2. The Random moving Prey can operate in GridWorld Environments using a specialized Interaction class.

4.2 Design and Implementation

As a first step in constructing a multi-agent system for the pursuit problem, general classes for agents in gridworlds have been designed and implemented. This approach was taken with a view on eventual future implementations of gridworld problems other than the pursuit problem. Next, a pursuit problem specific environment and interaction have been designed and implemented. Again, this was quite straightforward, so no further discussion is required.

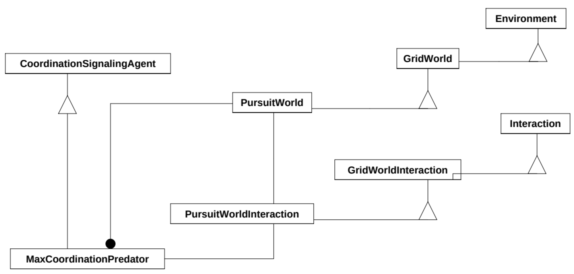

Fig. 3. The MaxCoordinationPredator can operate in PursuitWorld Environments and sends *evaluation signals* to its neighbors.

4.3 Coordination Signals

As mentioned in the general discussion on instantiating the framework, an important step in making an instantiation of the Coordination Signal Framework is the definition of application-specific coordination signals. These signals will in future work be used to teach new, domain-independent agents to learn to choose the right action in each situation.

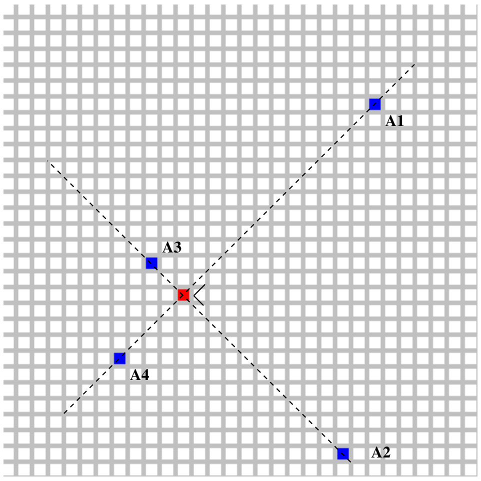

Fig. 4. Result of a high spread factor: optimal angles between predators are favored over distance minimization.

Figure 4 shows a prey surrounded by four predators, $A_1..A_4$. A CoordinationSignalingAgent sends evaluations of the coordination between itself and its left and right neighbors. Consequently, it also receives a coordination evaluation from its left and right neighbors. The MaxCoordinationPredators chooses its moves by maximizing the sum of the two evaluations it is given. In the figure, A1 has neighbors A2 and A3, and will therefore move to the position that maximizes

$$Eval(A_2, A_1) + Eval(A_3, A_1)$$

It should be noted that in some cases, the agent will have different neighbors at a position it is investigating as a possible choice to move to. As coordination evaluation in the pursuit problem was chosen to apply to direct neighbors only, the agent will take into account the coordination evaluation of this new neighbor, and not of its neighbor at the current position.

In the Pursuit Problem domain, two factors appear to be important in surrounding a prey: moving towards it, and surrounding it. The coordination evaluation therefore combines a distance factor and a spread factor.

The distance factor should encourage minimizing the distance between a predator and a prey. Just minimizing the distance to the prey can be done by any individual, and requires no coordination. To coordinate moving towards the prey, we combine this factor with the degree to which an agent and his neighbor are at the same distance of the prey; the idea is that they should gradually approach the prey. All evaluation factors are in the interval [0..1], 0 representing a bad evaluation and 1 a perfect one.
The Equidistance parameter determines the weight that is attributed to equidistance relative to the distance factor. E.g. an equidistance parameter of 1 does not take absolute distance into account, only the degree to which the distances of the agent and his neighbor are similar, and one of 0.5 would take both factors into account evenly. An agent's position is expressed in polar coordinates relative to the prey.
The spread factor can be formalized by means of a factor that takes the angles of the predators relative to the prey into account. When the agents are maximally spread, the angles between them are equal to

$$\phi_{opt} = \frac{2\pi}{\#predators}$$

The overall coordination evaluation signal is a single value that combines the distance factor and the spread factor.

$$\begin{aligned}
Eval(A_p, A_q) &= \delta \cdot Dist(A_p, A_q) \ + \ (1-\delta) \cdot Spread(A_p, A_q) \\
Dist(A_p, A_q) &= \epsilon \cdot equidistance(A_p, A_q) \ + \ (1-\epsilon) \cdot distance(A_p, A_q) \\
equidistance(A_p, A_q) &= 1 - \frac{|A_p.d - A_q.d|}{max(A_p.d, A_q.d)} \\
distance(A_p, A_q) &= 1 - \frac{A_p.d + A_q.d}{2r\sqrt{2}} \\
Spread(A_p, A_q) &= 1 - \frac{|\phi_{opt} - ((A_p.phi - A_q.phi) \ mod \ 2\pi)|}{E_{max}} \\
&where \\
E_{max} &= 2\pi - \phi_{opt}
\end{aligned}$$

For an agent A, $A.\phi$ is the angle and $A.d$ the distance, both relative to the prey. δ is the distance parameter (weight of $Dist$, relative to $Spread$). ϵ is the equidistance parameter (weight of the equidistance factor, relative to the distance factor). E_{max} is the maximal error in the angle. The MaxCoordination-Predator makes its choices using information that is locally determined, with a global view. This implies that no intentions of other agents (nor information in which these are implicit) are known to the agent. It is therefore difficult to locate the communication complexity in or between the three control systems of [Stephens and Merx, 1990] that have been described; the MaxCoordination-Predator has no knowledge of intentions of other agents, which would put it on a par with their *local control*. However, it *does* have a global view which, as a matter of fact, it uses only to determine the positions of its two neighbors. It seems difficult to compare the value of this information to the value of knowledge of other agents' intentions.

5 Results

The coordination evaluation formula contains two parameters. These had to be fixed first. The Equidistance parameter and the distance parameter were both initially set to zero. With these parameters, the agents judge the coordination with their neighbors to be useful when they maximize their spreading around the prey. This yields no prey-following behavior, but causes the agents to try to maintain angles of 90 degrees. Then the distance parameter was gradually increased. The Equidistance parameter was kept at 0, causing the degree to which predators are at comparable distances to the prey not to be taken into account. Even at low values of the distance parameter (e.g. 0.05), the predators move towards the random moving prey. An interesting anomaly occurred: sometimes, the agents would converge to the *orthogonal* positions around the prey (i.e. left, right, above and below it), but in many cases they converged to the diagonal positions.

As the angles between the agents are also optimal in this situation, it is another stable configuration. However, in the 4-connected pursuit problem we are investigating, only the first of these two stable positions is a capture situation. This implies that the coordination evaluation formula should favor the first configuration over the second. It turned out that raising the distance parameter yields just that result, which is achieved at a distance parameter value of 0.96. For more details, see [De Jong, 1997]. With these parameters, we tested the performance of the system. This turned out to be satisfactory for our purposes. Further experimentation with the equidistance parameter did not yield better results.

5.1 Analysis of Performance

Figure 5 shows how the constituent factors of the coordination evaluation for one predator evolve over time during one pursuit. Note that the best results were obtained with a distance parameter of 0.96, which results in a coordination

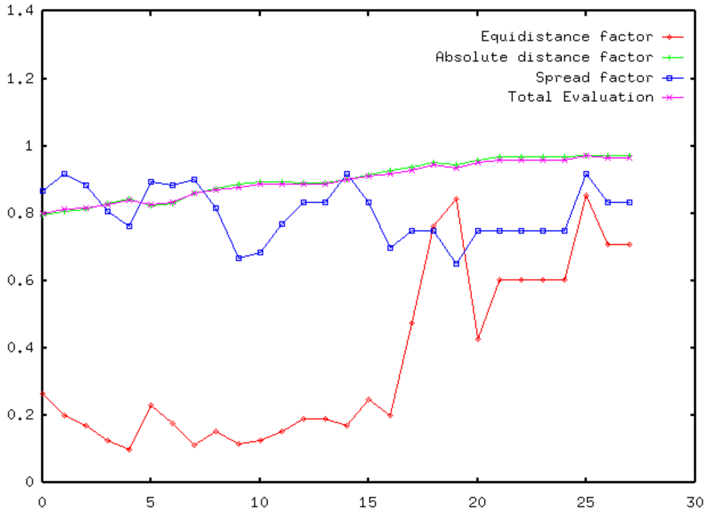

Fig. 5. Development of the evaluation's constituent factors for one predator over time in one pursuit.

evaluation approaching the distance term *Dist* very closely. The distance factor reflects the absolute distance because the equidistance parameter is set to zero.

Figure 6 shows the total coordination evaluations as received by the 4 predators over time in one pursuit. It rises gradually, and stays just below its theoretical maximum of 1. This maximum cannot be attained because the predators are not allowed to move over the prey, and consequently the distance term never reaches the optimum of 1.

Our goal was not to construct an optimal predator for pursuit problems, but to test if it is possible to define a coordination evaluation factor that, when maximized, is acceptable in this testbed domain (i.e. yields reasonable predator behavior). If this turns out not to be the case, then the idea of teaching a CoordinationLearningAgent using this factor would have to be rethought.

	Capture	Stalemate	Escape
Local-control[1]	10	47	43
Distributed-control[1]	83	13	3
Central-control[1]	100	0	0
MaxCoordinationPredator	100	0	0

Table 1. Pursuit outcomes (percentages) of Stephens & Merx (30 trials) and MaxCoordinationPredator (100 trials)

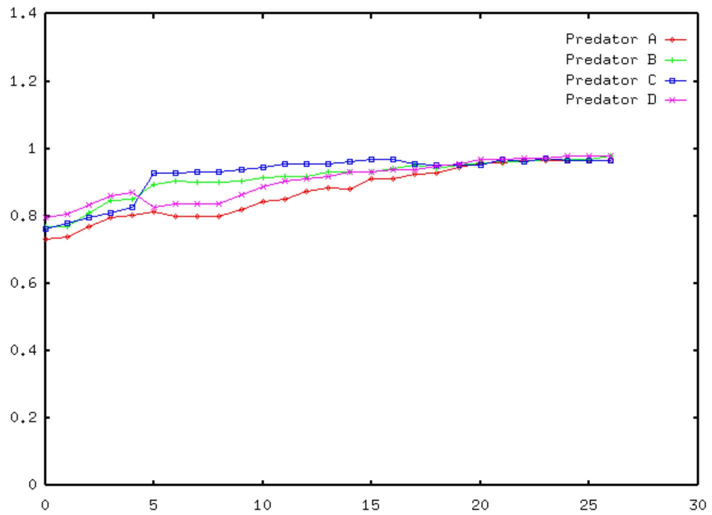

Fig. 6. Development of the total evaluation for four predators over time in one pursuit.

The first table shows the relative number of captures, stalemates and escapes. The MaxCoordinationPredators yielded the same results as central control, i.e. perfection.

	Capture ratio	Success ratio	Success efficiency
Local-control[1]	0.100	0.333	0.319
Distributed-control[1]	0.833	0.900	0.697
Central-control[1]	1.000	1.000	0.641
MaxCoordinationPredator	1.000	1.000	0.667

Table 2. Performance metrics of Stephens & Merx (30 trials) and MaxCoordination-Predator (100 trials)

The second table shows the efficiency of the predators. As all games resulted in a capture for both the MaxCoordinationPredator, the capture ratio and success ratio are not of interest. The success efficiency is just between Stephens and Merx's two most successful strategies. A game theoretic approach to the pursuit problem encountered in the literature (see [Levy and Rosenschein, 1992]) does not yield better results. In [Korf, 1992], an elegant solution to pursuit games is given. For hexagonal and diagonal (8-connected) games, a distance factor com-

[1] Stephens & Merx

bined with a repulsion factor yielded very successful results. The repulsion factor is based on distances between predators, and has a function comparable to that of our spread factor, which is based on angles between predators relative to the prey. A difference is that our spread factor influenced results positively in the orthogonal (4-connected) games we are concerned with here, whereas Korf found the repulsion factor to cause stalemates in orthogonal games and therefore only applied it in the hexagonal and diagonal variants. Other differences with our experiments are that the grid is 100 x 100 instead of 30 x 30, and that the prey moves to the position that is furthest away from the nearest predator and rests in 10% of its moves, instead of randomly choosing between the allowed options. These differences make a comparison of the methods difficult; it is reasonable to assume a random moving prey can be captured more easily than one that evades its predators. For the orthogonal (4-connected) case, Korf used a distance function based on the *max norm* which, like his other solution and like ours, resulted in a capture in all cases. The results were less satisfactory than those of the other two types of game, since the system had to be stopped artificially when a capture occurred to prevent further motion of the predators. Our solution did not suffer from this problem.

The coordination evaluation factor can be concluded to be more than sufficient for our purposes. We therefore conclude that the concept of a domain-independent CoordinationLearningAgent, which learns to act by relating a CoordinationSignalingAgent's signals to its interaction with the environment is, at least in the case of the pursuit problem, in principle possible.

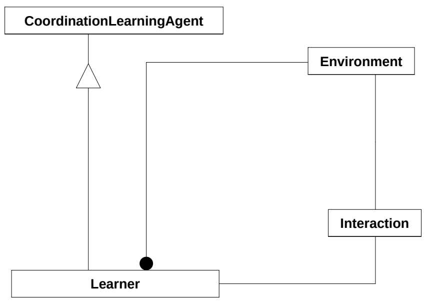

Fig. 7. A Learning Agent can act in any environment by learning to maximize the evaluations it receives.

6 Conclusions

A framework for coordination in multi-agent systems has been described. The goal of the experiments that are reported on here, was to acquire experience with the framework's instantiation to a domain, and moreover to test whether it is possible to base an agent's behavior on the coordination signals that have

been defined.
Both findings were positive. The instantiation of the subclasses for a multi-agent system was straightforward. The MaxCoordinationPredator, an agent that chooses its actions by maximizing the coordination evaluation signals it will receive, ranks among the best predators found in literature on the pursuit problem.

7 Future work

If coordination learning agents in a finite domain, such as the pursuit problem, would have a perfect memory and infinite experience, they would be able to perform just as good as agents that were designed specially for that domain. Two factors trouble this ideal picture. One is that in many domains, waiting until all situations have appeared in the learning phase is clearly unacceptable; even in a simple problem as the pursuit problem the number of different situations that can occur is large (in a 30 x 30 grid $900!/895! \approx 5.8 \cdot 10^{14}$, assuming the individual agents can be recognized). The other one is that machine learning algorithms have not reached the state of perfection. Because of the large number of possible situations, learning will have to depend on features that capture the relevant aspects of a domain, and automatic feature extraction remains a hard task. Nevertheless, in the case study of the framework's application to the pursuit problem, acting based on the coordination evaluation signals turned out to yield satisfactory results. In future research, we want to investigate CoordinationLearningAgents, who learn to relate coordination evaluation to the situation and thus learn to be useful in pursuing a prey, even though not designed for this task. Possible extensions are using multiple inheritance (agents that both send signals and learn from signals), extending the vocabulary (enabling agents to suggest actions to other agents), and using non-random preys.

8 Acknowledgements

The author wants to thank Bram Bakker, Dick de Ridder and 3 anonymous reviewers for useful comments on an earlier version, and Walter van de Velde and Ronald Schrooten for interesting and useful discussions. This research was partially funded by EU Telematics Information Engineering projects MAGICA and GEOMED. GEOMED (IE 2037) stands for 'Geographical mediation systems'. MAGICA (IE 2069) stands for 'Multi-Media Agent-based Interactive Catalogs'. Additional funding has been provided by an OZR contract for research on *models for coordination and competition in multi agent systems.*

References

[Benda et al., 1988] Benda, M., Jagannathan, V., and Dodhiawalla, R. (1988). On optimal cooperation of knowledge sources. Technical Report BCS-G2010-28, Boeing AI Center.

[Brazdil et al., 1991] Brazdil, P., Gams, M., Sian, S., Torgo, L., and Van de Velde, W. (1991). Learning in distributed systems and multi-agent environments. In Kodratoff, Y., editor, *Machine learning - EWSL-91*, Lecture Notes in Artificial Intelligence, vol. 482, Berlin. Springer Verlag.

[De Jong, 1997] De Jong, E. (1997). Ai-memo 97-05. Technical report, Vrije Universiteit Brussel, AI Lab.

[Genesereth et al., 1986] Genesereth, M. R., Ginsberg, M. L., and Rosenschein., J. S. (1986). Cooperation without communication. In *Proceedings of the National Conference on Artificial Intelligence*, Philadelphia, Pennsylvania. AAAI-86.

[Korf, 1992] Korf, R. E. (1992). A simple solution to pursuit games. In *Working Papers of the 11th International Workshop on Distributed Artificial Intelligence.*

[Kraus and Rosenschein, 1992] Kraus, S. and Rosenschein, J. (1992). The role of representation in interaction: Discovering focal points among alternative solutions. *Decentralized Artificial Intelligence III.*

[Levy and Rosenschein, 1992] Levy, R. and Rosenschein, J. (1992). A game theoretic approach to distributed artificial intelligence and the pursuit problem. *Decentralized Artificial Intelligence III.*

[M. Fenster and Rosenschein, 1995] M. Fenster, S. K. and Rosenschein, J. (1995). Coordination without communication: Experimental validation of focal point techniques. In *ICMAS '95.*

[Maturana and Varela, 1992] Maturana, H. and Varela, F. (1992). *The Tree of Knowledge: The biological roots of human understanding.* Shambala, Boston and London.

[Minsky, 1985] Minsky, M. (1985). *The Society of Mind.* Simon & Schuster, London.

[Rumbaugh et al., 1991] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., and Lorensen, W. (1991). *Object-Oriented Modeling and Design.* Prentice-Hall International, Inc.

[Sandip Sen and Hale, 1994] Sandip Sen, M. S. and Hale, J. (1994). Learning to coordinate without sharing information. In *Proceedings of the National Conference on Artificial Intelligence*, pages 426–431, Seattle, Washington.

[Sen and Sekaran, 1995] Sen, S. and Sekaran, M. (1995). Multiagent coordination with learning classifier systems. In G. Weiss, S. S., editor, *Adaption and Learning in multi-agent systems*, pages 218–233, Berlin, Heidelberg. Springer Verlag.

[Stephens and Merx, 1990] Stephens, L. M. and Merx, M. B. (1990). The effect of agent control strategy on the performance of a dai pursuit problem. In *Proceedings of the 1990 Distributed AI Workshop.*

[Van de Velde, 1996] Van de Velde, W. (1996). Cognitive architectures - from knowledge level to structural coupling. In Steels, L., editor, *The Biology and Technology of Intelligent Autonomous Agents*, pages 197–221.