Learning to Coordinate Actions in Multi-Agent Systems

Gerhard WeiB
Institut fur Informatik, Technische Universitat Miinchen
Arcisstr. 21, 8000 Miinchen 2, Germany
weissg@informatik.tu-muenchen.de

Abstract

This paper deals with learning in reactive multi-agent systems. The central problem addressed is how several agents can collectively learn to coordinate their actions such that they solve a given environmental task together. In approaching this problem, two important constraints have to be taken into consideration: the incompatibility constraint, that is, the fact that different actions may be mutually exclusive; and the local information constraint, that is, the fact that each agent typically knows only a fraction of its environment.
The contents of the paper is as follows. First, the topic of learning in multi-agent systems is motivated (section 1). Then, two algorithms called ACE and AGE (standing for "ACtion Estimation" and "Action Group Estimation", respectively) for the reinforcement learning of appropriate sequences of action sets in multi agent systems are described (section *2).* Next, experimental results illustrating the learning abilities of these algorithms are presented (section 3). Finally, the algorithms are discussed and an outlook on future research is provided (section 4).

1 Introduction

Multi-Agent Systems. In computer science and artificial intelligence the concept of multi-agent systems has influenced the initial developments in areas like cognitive modelling [Selfridge, 1959; Minsky, 1979], blackboard systems [Erman and Lesser, 1975], object-oriented programming languages [Hewitt, 1977], and formal models of concurrency [Petri, 1962; Brauer *et ai,* 1987]. Nowadays multi-agent systems establish a major research subject in distributed artificial intelligence; see [Bond and Gasser, 1988; Brauer and Hernandez, 1991; Gasser and Huhns, 1989; Huhns, 1987]. The interest in multi-agent systems is largely founded on the insight that many real-world problems are best modelled using a set of agents instead of a single agent. In particular, multi-agent modelling makes it possible *(i)* to cope with natural constraints like the limitation of the processing power of a single agent or the physical distribution of the data to be processed and *(it)* to profit from inherent properties of distributed systems like robustness, fault tolerance, parallelism and scalability.

Generally, a multi-agent system is composed of a number of agents that are able to interact with each other and the environment and that differ from each other in their skills and their knowledge about the environment. (Usually an individual agent is assumed to consist of sensor component, a motor component, a knowledge base, and a learning component.) There is a great variety in the multi-agent systems studied in distributed artificial intelligence [Huhns, 1987, foreword]. This paper deals with *reactive* multi-agent systems, where 'reactive" means that the behavior and the environment of the system are strongly coupled (there is a continuous interaction between the system and its environment).

Learning. There is a common agreement that there are two important reasons for studying learning in multi-agent systems: to be able to endow artificial multi-agent systems (e.g., systems of interacting autonomous robots) with the ability to automatically improve their behavior; and to get a better understanding of the learning processes in natural multi-agent systems (e.g., human groups or societies). In a multi-agent system two forms of learning can be distinguished [Shaw and Whinston, 1989]. First, *centralized* or *isolated learning,* i.e. learning that is done by a single agent on its own (e.g. by creating new knowledge structures or by practicing motor activities). And second, *distributed* or *collective learning,* i.e. learning that is done by the agents as a group (e.g. by exchanging knowledge or by observing other agents). This paper focusses on collective learning, and the central question addressed is: "How can eacn agent learn which action it shall perform under which circumstances?" In answering this question, two important constraints have to be taken into consideration [WeiB, 1993a, 1993b]. First, the *incompatibility constraint,* i.e. the fact that different actions may be incompatible in the sense that the execution of one action leads to environmental changes that impair or even prevent the execution of the others. And second, the *local information constraint,* i.e. the fact that an agent typically has only local information about the actualenvironmental state, and this information may differ from the one another agent has; this situation is illustrated by figure 1.

Two algorithms called the *ACE algorithm* and the *AGE algorithm* for reinforcement learning in reactive multi-agent systems are described (ACE and AGE are acronyms for "ACtion Estimation" and "Action Group Estimation", respectively). These algorithms base on the action-oriented version [WeiB, 1992] of the bucket brigade learning model for classifier systems [Holland, 1986]. According to both algorithms the agents collectively learn to estimate the goal relevance of their actions and, based on their estimates, to coordinate their actions

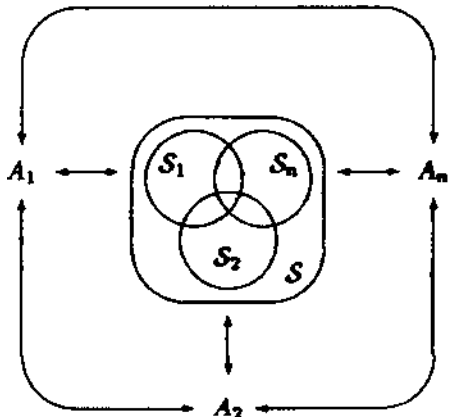

Figure 1: System—environment interaction. Each agent A_i only knows a fraction S_i of the actual environmental state $\mathcal{S}$. The agents may know different aspects of the actual state, and there may be environmental aspects that none of the agents knows. The agents interact with each other as well as with the environment.

and to generate appropriate action sequences.

Notational Preliminaries. For a description of the ACE and the AGE algorithm the following elementary notation is used throughout the rest of this paper. $\mathcal{A}g = \{A_1, \ldots, A_n\}$ $(n \in \mathbb{N})$ denotes the set of all agents. $\mathcal{S}$ $(T, U, \ldots)$ refers to an environmental state, and S_i refers to the part of $\mathcal{S}$ that is known to the agent $A_i \in \mathcal{A}g$ $(S_i \subseteq \mathcal{S})$; S_i is called A_i*'s knowledge about* $\mathcal{S}$. $Ac_i^{poss} = \{a_i^1, \ldots, a_i^{m_i}\}$ $(m_i \in \mathbb{N})$ denotes the set of all possible actions of the agent A_i, and it is called the *action potential* of A_i. $Ac_i^{poss}[S]$ denotes the set of all actions that A_i could carry out (identifies as "executable") in the environmental state $\mathcal{S}$ $(Ac_i^{poss}[S] \subseteq Ac_i^{poss})$.

It is worth emphasizing that $S_i \cap S_j$ may but need not be empty, and that $\bigcup_{i=1}^n S_i$ is not necessarily equal to $\mathcal{S}$ (i.e. the agents may have incomplete information about an environmental state, see figure 1). Similarly, if $\mathcal{S}$ and $\mathcal{T}$ are two different environmental states, then $S_i \cap T_i$ may but need not be empty; in particular, it may be the case that $S_i = T_i$ (i.e. an agent may be unable to distinguish between different environmental states).

2 Collective Learning

The ACE Algorithm. According to the ACE algorithm the learning activity of the multi-agent system results from the repeated execution of a basic *working cycle* which consists of the three activitities *action determination*, *competition*, and *credit assignment* as follows.

First, each agent $A_i \in \mathcal{A}g$ determines, in dependence on its knowledge S_i about the actual environmental state $\mathcal{S}$, the set $Ac_i^{poss}[S]$ of actions that it could carry out.

Then the agents compete for the right to become active. This competition encompasses the calculation and announcement of bids and the selection of the actions that are actually carried out:

(i) Each agent A_i makes a bid $B_i^j[S]$ for each of its possible actions $a_i^j \in Ac_i^{poss}[S]$ and announces it to the other agents. The bid is calculated by

$$B_i^j[S] = \begin{cases} (\alpha + \beta) \cdot E_i^j[S] & : E_i^j[S] > \Theta \\ 0 & : \textit{otherwise} \end{cases}, \quad (1)$$

where α is a small constant called *risk factor*, β is a small random number called *noise term*, Θ is a constant called *estimate minimum*, and $E_i^j[S]$ is A_i's estimate of the goal relevance of a_i^j in dependence on its knowledge S_i about $\mathcal{S}$. The α indicates the fraction of $E_i^j[S]$ the agent A_i is willing to risk for being allowed to perform the action a_i^j; the β introduces noise into the competition in order to avoid getting stuck into local learning minima; and the Θ helps to prevent executing useless (low-estimated) actions. (Whenever an agent A_i can execute an action under some knowledge S_i for the first time, it initializes the estimated goal relevance of this action with a predefined value E^{init}; afterwards the estimated goal relevance is adjusted during credit assignment as it is described below.)

(ii) After the agents have announced their bids, they select the actions to be carried out. The agent having made the highest bid is allowed to execute its corresponding action, and all agents withdraw the bids for those actions that are incompatible to this selected action; this is repeated until no further action being associated with a non-zero bid can be selected. This action selection may be formally described as follows:

- $Ac^{poss}[S] = \bigcup_{i=1}^n Ac_i^{poss}[S]$ and $\mathcal{A} = \emptyset$
- until $Ac^{poss}[S] = \emptyset$ do
  - select $a_i^j \in Ac^{poss}[S]$ with $B_i^j[S] \geq B_k^l[S]$ for all $a_k^l \in Ac^{poss}[S]$
  - $\mathcal{A} = \mathcal{A} \cup \{a_i^j\}$
  - $Ac^{poss}[S] = Ac^{poss}[S] \setminus (\, a_i^j \cup \{a_k^l \in Ac^{poss}[S] : a_k^l$ and a_i^j are incompatible$\}$)

The set $\mathcal{A}$ is called the actual *activity context*, and the actions contained in in this set are called the *actual actions* (in state $\mathcal{S}$). (The selection requires a rational or non-egoistic behavior of the agents in the sense that none of the agents insists the execution of a low-bid or an incompatible action.)

Finally, the agents assign credit to each other by adjusting the estimates of the goal relevance of their actions. This adjustment is done according to the action-oriented bucket brigade mechanism [Weiß, 1992]. Informally, the agents reduce the estimates of their actual actions (the actual winners pay for their previlege to carry out their actions), and hand the amount of all reductions back to the agents that won the previous competition (the previous winners are rewarded for appropriately setting up the environment). The previous winners, in turn, add the received amount to the estimates of the actions they performed last. Additionally, if there is an external reward from the environment, then the agents distribute this reward among the actual actions. Formally, this can be described as follows:

(i) For each actual action $a_i^j \in \mathcal{A}$ the agent A_i modifies its estimate $E_i^j[S]$ according to

$$E_i^j[\mathcal{S}] = E_i^j[\mathcal{S}] - B_i^j[\mathcal{S}] + R^{ext}/|\mathcal{A}| \ , \qquad (2)$$

where $B_i^j[\mathcal{S}]$ is the corresponding action-specific bid and R^{ext} is the external reward (if there is any).

(ii) The agents sum up the bids they made for the actual actions. The resulting sum $B_{\mathcal{A}}[\mathcal{S}]$,

$$B_{\mathcal{A}}[\mathcal{S}] = \sum_{a_i^j \in \mathcal{A}} B_i^j[\mathcal{S}] \ , \qquad (3)$$

is distributed in equal shares among those actions that were carried out during the previous competition. Suppose that $\mathcal{T}$ is the previous environmental state, $\mathcal{B}$ is the previous activity context, and $a_k^l \in \mathcal{B}$. Then $E_k^l[\mathcal{T}]$ is increased by

$$E_k^l[\mathcal{T}] = E_k^l[\mathcal{T}] + B_{\mathcal{A}}[\mathcal{S}] \,/\, |\mathcal{B}| \ . \qquad (4)$$

This credit assigment mechanism has two major effects [Holland, 1985]: the estimates of actions that are involved in a successful sequence of action sets (i.e. a sequence that leads to external reward) increase over time and, by the way, stabilize this sequence; and conversely, the estimates of actions that are involved in an unsuccessful sequence decrease over time and destabilize this sequence.

The AGE Algorithm. The AGE algorithm retains the basic working cycle of the ACE algorithm — repeated execution of action determination, competition and credit assignment — but realizes competition in a different way. Now an agent estimates the goal relevance of an action not only in dependence on its knowledge about the actual environmental state but also in dependence on the *possible activity contexts*, and the agents do not compete for carrying out individual actions but for carrying out *groups of actions*. Generally, an activity context is a group of mutually compatible actions. Formally, the set of all possible activity contexts in an environmental state $\mathcal{S}$ is given by

$$\mathcal{A}[\mathcal{S}] = \{\mathcal{A} \subseteq \bigcup_{i=1}^{n} Ac_i^{poss}[\mathcal{S}]: \\ (\forall a_k^l, a_p^q \in \mathcal{A}: a_k^l \text{ and } a_p^q \text{ are compatible})\} \ . \qquad (5)$$

The competition is organized as follows (let $\mathcal{S}$ be the actual environmental state).

(i) For each $\mathcal{A} \in \mathcal{A}[\mathcal{S}]$ the agents calculate a bid $B_{\mathcal{A}}[\mathcal{S}]$ for being allowed to carry out *all* the actions contained in $\mathcal{A}$ by

$$B_{\mathcal{A}}[\mathcal{S}] = \sum_{a_i^j \in \mathcal{A}} B_i^j[\mathcal{S}, \mathcal{A}] \qquad (6)$$

with

$$B_i^j[\mathcal{S}, \mathcal{A}] = (\alpha + \beta) \cdot E_i^j[\mathcal{S}, \mathcal{A}] \ , \qquad (7)$$

where α is a risk factor, β is a noise term, and $E_i^j[\mathcal{S}, \mathcal{A}]$ is A_i's estimate of the goal relevance of a_i^j in dependence on $\mathcal{S}_i$ and $\mathcal{A}$.

(ii) The activity context $\mathcal{A} \in \mathcal{A}[\mathcal{S}]$ being associated with the greatest bid $B_{\mathcal{A}}[\mathcal{S}]$ is selected, and all actions contained in this context are carried out.

Credit assignment is done analogously to the ACE algorithm (suppose that the activity context $\mathcal{A}$ has been selected):

(i) For each $a_i^j \in \mathcal{A}$ the agent A_i modifies its estimate $E_i^j[\mathcal{S}, \mathcal{A}]$ according to

$$E_i^j[\mathcal{S}, \mathcal{A}] = E_i^j[\mathcal{S}, \mathcal{A}] - B_i^j[\mathcal{S}, \mathcal{A}] + R^{ext}/|\mathcal{A}| \ . \qquad (8)$$

(ii) The total bid $B_{\mathcal{A}}[\mathcal{S}]$ is distributed among the actions that were carried out in the previous competition. Suppose that $\mathcal{T}$ is the previous environmental state and that $\mathcal{B}$ is the previously selected activity context. For each $a_k^l \in \mathcal{B}$ the agent A_k increases $E_k^l[\mathcal{T}, \mathcal{B}]$ by

$$E_k^l[\mathcal{T}, \mathcal{B}] = E_k^l[\mathcal{T}, \mathcal{B}] + B_{\mathcal{A}}[\mathcal{S}] \,/\, |\mathcal{B}| \ . \qquad (9)$$

Like the ACE-type adjustment does, this AGE-type adjustment of the estimates induces the stabilization of successful sequences of action sets and to the destabilization of unsuccessful ones.

3 Experimental Results

Task Domain. As a task domain the blocks world is chosen. This domain is clearly enough for experimental studies in an unknown field like collective learning, and it is well suited for illustrating the essential features of the ACE algorithm and the AGE algorithm.

The task to be solved by the agents is to transform a start configuration of blocks into a goal configuration within a limited time interval. Figure 2 shows such a task together with the actions that can be carried out by the agents. In this example each agent is assumed to have limited motor capabilities and to be specialized in moving one specific block; for instance, agent A_1 is responsible for block A, and is able to put this block on block B ($put(A, B)$) or on the ground ($put(A, \perp)$). The precondition for the application of an action $put(x, y)$ is that the blocks x and y are empty, i.e. that no other blocks are placed on them. (Note that an agent's action is quite complex and involves a number of activities; for instance, in order to put A on B the agent has to walk to A, to pick up A, to walk to B, and to place A on B.)

Each agent is assumed to have limited sensor capabilities (it only "sees" what is directly relevant to its actions) and, as a consequence, to have only local information about each environmental state. The part $\mathcal{S}_i$ of an environmental state $\mathcal{S}$ that is known to an agent A_i is specified as follows. For each of its actions $a_i^j \equiv put(x, y)$ an agent only knows *(i)* the block on which $z \in \{x, y\}$ is positioned and *(ii)* whether $z \in \{x, y\}$ is empty. The knowledge specified by *(i)* and *(ii)* is called the *environmental context* of the action $put(x, y)$.

Furthermore, two actions are considered to be incompatible if the execution of one of them changes the environmental context of the other. Formally, this can be expressed as follows: two actions, $put(x, y)$ and $put(u, v)$, are incompatible if $x \in \{u, v\}$ or $u \in \{x, y\}$ or $y = v \neq \perp$. Examples of sets of incompatible actions are $\{put(B, C), put(B, D)\}$ ("a block cannot be put on different blocks at the same time"), $\{put(D, C), put(E, C)\}$ ("two different blocks cannot be put on the same block at the same time"), and $\{put(E, B), put(B, C)\}$ ("a block cannot be put on a block which at the same time is put on another block"). It is assumed that the agents know these incompatibility constraints.

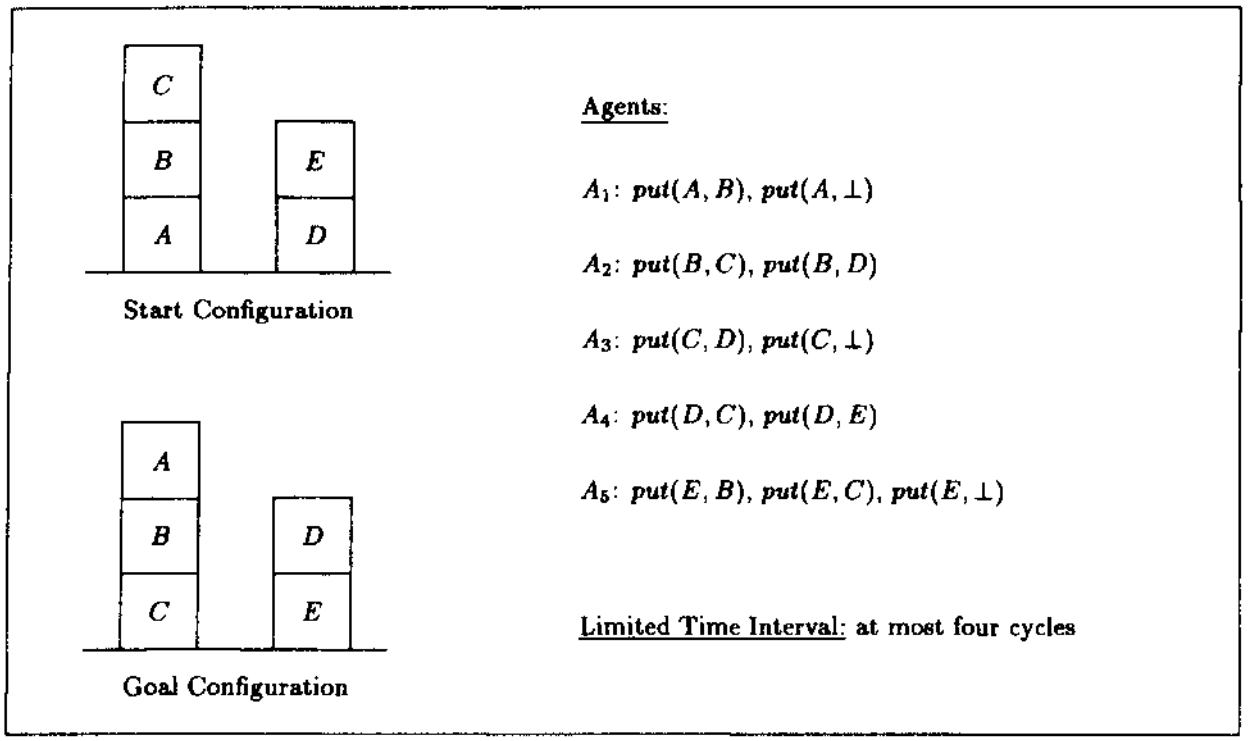

Figure 2: A blocks world task.

As it is described in section 2, learning proceeds by the repeated execution of the basic working cycle. A *trial* is defined as any sequence of at most four cycles that transforms the start into the goal configuration (successful trial), as well as any sequence of exactly four cycles that transforms the start into a non-goal configuration. At the end of each trial the start configuration is restored, and it is again presented to the agents. Additionally, at the end of each successful trial a non-zero external reward R^{ext} is provided.

Task Analysis. As a consequence of the local information constraint, an agent may be unable to distinguish between environmental states in which its actions are useful and relevant to goal attainment and environmental states in which its actions are useless. (This situation is sometimes called the *Sussman anomaly.)* Consider the environmental states T, *U* and V shown in figure 3. Based on the usual blocks world notation, these three states are completely described by

$$\begin{aligned} T &= \{on(A,\perp), on(B,C), on(C,\perp), on(D,E), \\ &\quad on(E,\perp), empty(A), empty(B), empty(D)\}\ , \\ U &= \{on(A,\perp), on(B,D), on(D,\perp), on(C,\perp), \\ &\quad on(E,\perp), empty(A), empty(B), empty(C), \\ &\quad empty(E)\} \quad and \\ V &= \{on(A,\perp), on(B,C), on(C,D), on(D,\perp), \\ &\quad on(E,\perp), empty(A), empty(B), empty(E)\}\ . \end{aligned}$$

As it is easy to see, the action $put(A_1, B)$ of the agent A_1 is useful in state T but not useful in state V. However, because A_1's local information T_1 and V_1 about the states T and V, respectively, are identical, the agent A_1 is unable to distinguish between these two states. (Of course, an agent does not always fail to distinguish between "useful and useless states"; see e.g. the states *T* and *U*. A_1's local information is given by $T_1 = V_1 = \{on(A,\perp), on(B,C), empty(A), empty(B)\}$ and $U_1 = \{on(A,\perp), on(B,D), empty(A), empty(B)\}$.)

An analysis of the search space of the task depicted in figure 2 shows that there are only 3 successful trials of length 3, and 13 successful trials of length 4. The probability that a randomly generated sequence of applicable sets of compatible actions transforms the start into the goal configuration is 2.6 percent, if the sequence has the length 3, and 3.3 percent, if the sequence has the length 4. With that, the probability that a random trial solves the task to be learnt is less than 6 percent. (An example of a successful trial of length 3 is given by $\langle \{put(C,\perp), put(E,\perp)\}, \{put(B,C), put(D,E)\}, \{put(A,B)\} \rangle$. Note that a seguential "one-action-per-cycle" approach would require five cycles in order to implement this sequence.)

Experimental Results. A series of experiments was performed to test the ACE and the AGE algorithm. Figure 4 shows the performance profiles of the ACE algorithm, the AGE algorithm, and a random walk algorithm (i.e. an algorithm which randomly chooses an applicable set of compatible actions in each cycle). The parameter setting underlying these performance profiles was as follows: $E^{init} = R^{ext} = 1000, \alpha = 0.1, \beta \in [-\alpha/5 \ldots +\alpha/5]$ (randomly chosen), and $\Theta = 0.7 \cdot E^{init}$. (It has to be mentioned that the learning effects reported below can be observed for a broad range of parameters and are not limited to this setting.) Each data point in figure 4 reflects the average external reward per episode obtained during the previous 50 episodes. There are several important observations. First, the ACE and the AGE algorithm performed significantly better than the random walk algorithm, and they reached their maximum performance level after about 250 trials. After that, the performance levels remained almost constant; this shows that the ACE/AGE algorithms were able to learn stable sequences of action sets. Second, the perfor-

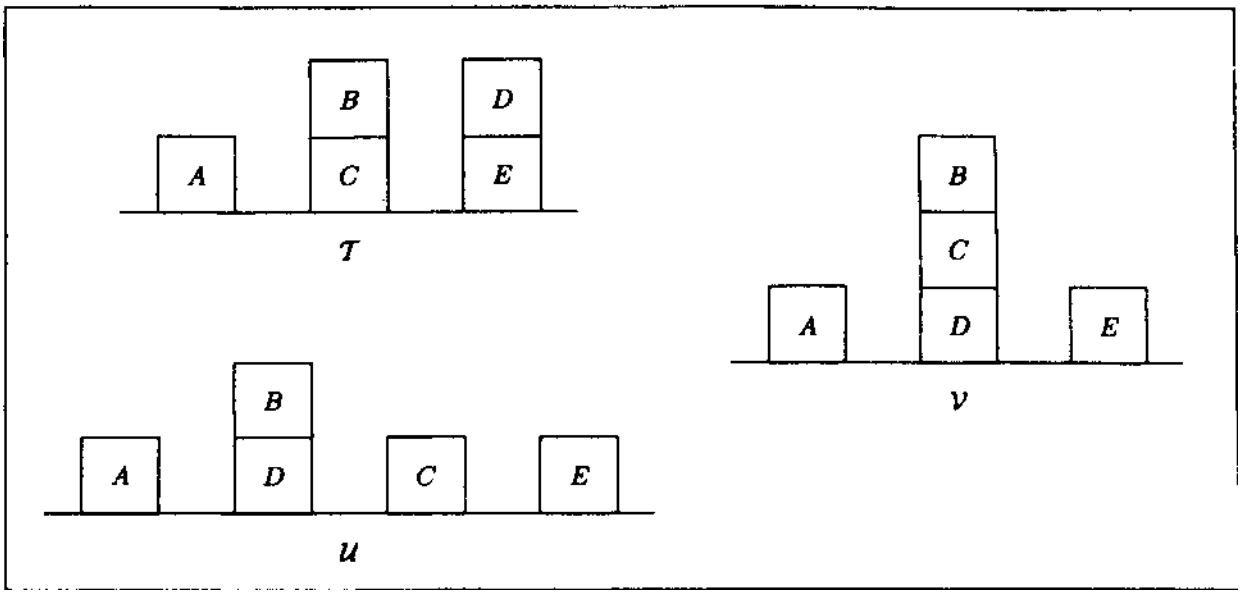

Figure 3: Blocks world states.

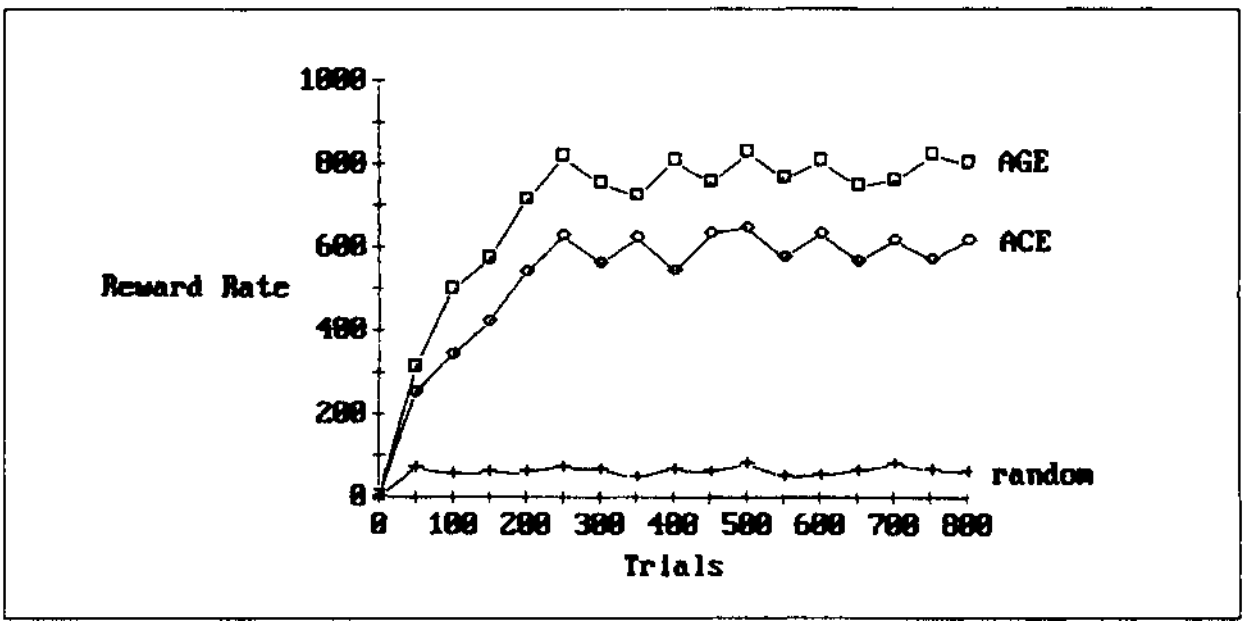

Figure 4: Performance profiles.

mance level of the AGE algorithm is clearly above the performance of the ACE algorithm. This illustrates the importance of estimating the goal relevance of an action, as it is done by the AGE algorithm, in dependence on other (concurrent) actions. The reason for that is that an action may be useful in one activity context but useless in another. However, the improved performance is achieved at the cost of higher space and computation time: whereas the costs of the ACE algorithm are proportional to the number of possible actions that can be carried out by the agents, the costs of the AGE algorithm are proportional to the number of possible action sets. And third, despite their learning abilities both algorithms remain below the possible maximal reward level (which is 1000). The reason for that is the local information constraint and, with that, the inability of the agents to distinguish between all different environmental states; as a consequence, the same estimates are used for different environmental states and necessarily remain inaccurate on some scale.

4 Concluding Remarks

This paper took the first steps towards learning to coordinate actions in multi-agent systems. Two algorithms called the ACE algorithm and the ACE algorithm for the delayed reinforcement learning of sequences of action sets were introduced and experimental results illustrating the learning abilities of these two algorithms were presented. Both algorithms are "elementary" in a twofold sense. On the one side, they make only weak demands on the cognitive abilities of the individual agents. For instance, they do not require that the agents are able to reason about the other agents' knowledge or intentions and they do not require that the agents possess complex decision making strategies. As a consequence, the algorithms are

even applicable to systems that are composed of rather simple agents. On the other side, both algorithms are very flexible learning schemes that can be extended in a number of ways. For instance, they allow to incorporate high-level problem solving and planning mechanisms known from the field of single-agent systems, as well as a number of refinements that have been proposed for the bucket brigade learning model (e.g., tax payment, support and look-ahead mechanisms).

Our future research will concentrate on these possible extensions of the ACE/AGE algorithms. A major topic is the development of algorithms that implement multi agent learning of sequences of compatible actions like the ACE/AGE algorithms do, but that better cope with the local information constraint.

Another goal of future research is the development of learning algorithms for more complex structured (e.g. hierarchically organized) multi-agent systems [Fox, 1981]. Up to now this topic has been not addressed in the field of distributed artificial intelligence. However, there are various related works from other disciplines like psychology (e.g., [Guzzo, 1982; Laughlin, 1988]) and economics (e.g., [Argyris and Schon, 1978; Galbraith, 1973; Hedberg, 1981; Sikora and Shaw, 1989J) that are likely to be very stimulating and useful for achieving this challanging goal.

References

[Argyris and Schon, 1978] C. Argyris and D.A. Schon. Organizational (earning. Addison Wesley. 1978.

[Bond and Gasser, 1988] A H. Bond and L. Gasser, editors. Readings in distributed artificial intelligence. Morgan Kaufmann. 1988.

[Brauer and Hernandez, 1991] W. Brauer and D. Hernandez, editors. Verteilte Kunstliche Intelltgenz und kooperatives Arbeiten. Springer. 1991.

[Brauer et a/., 1987] W. Brauer, W. Reisig, and R. Rozenberg, editors. Petri nets. Lecture Notes in Computer Science, Vol. 254 (Part I) and Vol. 255 (Part II). Springer. 1987.

[Erman and Lesser, 1975] L.D. Erman and V.E. Lesser. A multi-level organization for problem-solving using many, diverse, cooperating sources of knowledge. In Proceedings of the 1975 International Joint Conference on Artificial Intelligence (pp. 483-490). 1975.

[Fox, 1981] M.S. Fox. An organizational view of distributed systems. In IEEE Transactions on Systems, Man, and Cybernetics (Vol. SCM-11, No. 1, pp. 70-80). 1981.

[Galbraith, 1973] JR. Galbraith. Designing complex organizations. Addison Wesley. 1973.

[Gasser and Huhns, 1989] L. Gasser and M.N. Huhns, editors. Distributed artificial intelligence (Vol. 2). Pitman. 1989.

[Guzzo, 1992] R.A. Guzzo, editor. Improving group decision making in organizations - Approaches from theory and research. Academic Press. 1992.

[Hedberg, 1981] B. Hedberg. How organizations learn and unlearn. In P. C. Nystrom & W. H. Starbuck (Eds), Handbook of organizational design (Vol. 1, pp. 1-27). Oxford University Press. 1981.

[Hewitt, 1977] C.E. Hewitt. Viewing control structures as patterns of passing messages. Artificial Intelligence, 8(3), 323-364. 1977.

[Holland, 1985] J.H. Holland. Properties of the bucket brigade. In J. J. Grefenstette, editor, Proceedings of the First International Conference on Genetic Algorithms and Their Applications (pp. 1-7). 1985.

[Holland, 1986] J.H. Holland. Escaping brittleness: the possibilities of general-purpose learning algorithms to parallel rule-based systems. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), Machine learning: an artificial intelligence approach (Vol. 2, pp. 593-632). Morgan Kaufmann. 1986.

[Huhns, 1987] M.N. Huhns, editor. Distributed artificial intelligence. Pitman. 1987.

[Laughlin, 1988] P.R. Laughlin. Collective induction: group performance, social combination processes, and mutual majority and minority influence. Journal of Personality and Social Psychology, 54(2), 254-267. 1988.

[Minsky, 1979] M. Minsky. The society theory of thinking. In Artificial intelligence: an MIT perspective (pp. 423-450). MIT Press. 1979.

[Petri, 1962] C.A. Petri. Kommumkation mit Automates. Schriften des Instituts fur Instrumentelle Mathematik, Universitat Bonn, Germany. 1962.

[Selfridge, 1959] O.G. Selfridge. Pandemonium: a paradigm for learning. In Proceedings of the Symposium on Mechanisation of Thought Processes (pp. 511-529). Her Majesty's Stationery Office, London. 1959.

[Shaw and Whinston, 19891 M.J. Shaw and A.B. Whinston. Learning and adaptation in distributed artificial intelligence. In (Gasser & Huhns, 1989, pp. 413-429).

[Sian, 1990] S.S. Sian. The role of cooperation in multi-agent learning. In Proceedings of the First International Conference on Cooperating Knowledge Based Systems. 1990.

[Sian, 1991] S.S. Sian. Extending learning to multiple agents: issues and a model for multi-agent machine learning (MA ML). In Y. Kodratoff (Ed.), Machine learning — EWSL91 (pp. 440-456). Springer. 1991.

[Sikora and Shaw, 1990] R. Sikora and M. Shaw. A double-layered learning approach to acquiring rules for financial classification. Faculty Working Paper No. 90-1693, College of Commerce and Business Administration, University of Illinois at Urbana-Champaign. 1990.

[Weifi, 1992] G. Weifi. Learning the goal relevance of actions in classifier systems. In B. Neumann (Ed.), Proceedings of the 10th European Conference on Artificial Intelligence (pp. 430-434). Wiley. 1992.

[Weifi, 1993a] G. Weifi. Action selection and learning in multi-agent environments. Appears in Proceedings of the Second International Conference on Simulation of Adaptive Behavior. 1993.

[Weifi, 1993b] G. Weifi. Collective learning of action sequences. Appears in Proceedings of the 13th International Conference on Distributed Computing Systems. 1993.