Game-theoretic cooperativity in networks of self-interested units

Cite as: AIP Conference Proceedings **151**, 41 (1986); https://doi.org/10.1063/1.36249
Published Online: 05 June 2008

Andrew G. Barto

AIP | Conference Proceedings
Get 30% off all print proceedings!
Enter Promotion Code PDF30 at checkout

AIP Conference Proceedings **151**, 41 (1986); https://doi.org/10.1063/1.36249 **151**, 41

GAME-THEORETIC COOPERATIVITY IN NETWORKS OF SELF-INTERESTED UNITS

Andrew. G. Barto[1]
Department of Computer and Information Science
University of Massachusetts, Amherst MA 01003

ABSTRACT

The behavior of theoretical neural networks is often described in terms of competition and cooperation. I present an approach to network learning that is related to game and team problems in which competition and cooperation have more technical meanings. I briefly describe the application of stochastic learning automata to game and team problems and then present an adaptive element that is a synthesis of aspects of stochastic learning automata and typical neuron-like adaptive elements. These elements act as self-interested agents that work toward improving their performance with respect to their individual preference orderings. Networks of these elements can solve a variety of team decision problems, some of which take the form of layered networks in which the "hidden units" become appropriate functional components as they attempt to improve their own payoffs.

INTRODUCTION

The behavior of neural networks is often described in terms of competition, cooperation, and coalition formation [1,2,3,4]. Because these terms belong to the technical lexicon of game theory and economics, one is prompted to ask if they mean the same thing when used by network theorists? Certainly, as applied to networks, their ordinary meanings are preserved, but what of their more technical intent? Much of game theory and economics is based on the ideas of individual utility functions and utility maximization by self-interested agents (the assumption of individual rationality). These fields consider questions such as what social rationality should mean and how socially "optimal" behavior or resource allocation might be produced by interacting self-interested agents, where optimality is a far more complicated concept than it is in many other contexts. Competition arises due to conflicts of agent self-interest as embodied in the payoff structure of a game, and cooperation, that is, the formation of coalitions within which agents coordinate their activity, provides a means for agents to take advantage of overlapping interests.

To some extent, these basic ideas of game theory and economics have been faithfully interpreted in context of theoretical neural networks. For example, learning methods in which total synaptic strength is conserved (e.g., as in "competitive learning" [4]) might be regarded as including a resource allocation mechanism for a non-production economy, where the resource is the sum of the connection weights. Similarly, it may not be misleading to regard a unit in a Boltzmann machine [3] or a Hopfield network [5] as preferring to participate in low energy configurations with its neighbors.

Missing from these cases, however, are versatile agents that will attempt to make the best (according to their own interests) of whatever situation they may find themselves in. This requires agents whose decision strategies are not task-specific. In this paper I outline some aspects of our study of the collective behavior of self-interested neuron-like elements, an approach first suggested to us by the "hedonistic neuron" hypothesis of Klopf [6]. I begin by describing a type of adaptive agent that has been extensively studied in game and team decision problems.

[1] This research was supported by the Air Force Office of Scientific Research and the Avionics Laboratory (Air Force Wright Aeronautical Laboratories) through contract F33615-83-C-1078.

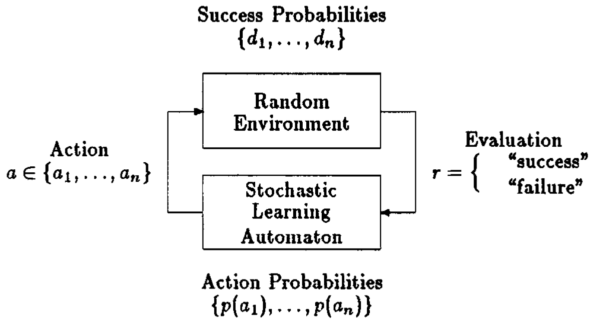
Success Probabilities
$\{d_1, \ldots, d_n\}$
Random Environment
Action
$a \in \{a_1, \ldots, a_n\}$
Evaluation
$r = \begin{cases} \text{"success"} \\ \text{"failure"} \end{cases}$
Stochastic Learning Automaton
Action Probabilities
$\{p(a_1), \ldots, p(a_n)\}$

Figure 1. Stochastic learning automaton interacting with a random environment.

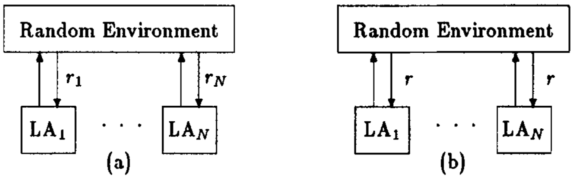

Figure 2. (a) The game problem. (b) The team problem.

STOCHASTIC LEARNING AUTOMATA

The theory of learning automata originated with the independent work of the Soviet cybernetician Tsetlin [7], mathematical psychologists studying learning, and statisticians studying sequential decision problems (e.g., the "n-armed bandit problem" [8]). Although this theory has an extensive modern literature in engineering (reviewed in ref. 9), it is unfortunate that there has been very little cross-fertilization between this theory and neural-network research.

Fig. 1 shows a learning automaton interacting with an environment. At each step in the processing cycle, the automaton randomly picks an action from a set of possible actions, $A = \{a_1, \ldots, a_n\}$, according to a vector of action probabilities, $P = \{p(a_1), \ldots, p(a_n)\}$. The environment then evaluates that action by selecting an evaluation signal that it transmits back to the automaton. Fig. 1 shows the case in which the evaluation, r, is either "success" or "failure" and is selected according to probabilities $\{d_1, \ldots, d_n\}$, where $d_i = \text{prob}\{\text{success}|a_i\}$ (other formulations allow a countable number or a bounded continuum of evaluations). Upon receiving the evaluation, the automaton updates its action probabilities as a function of its current action probabilities, the action chosen, and the environment's evaluation of that action. Beginning with no knowledge of the environmental success probabilities, the objective of the automaton is to improve its expectation of success over time. Ideally, it should eventually choose action a_l with probability 1, where $d_l = \max\{d_1, \ldots, d_n\}$. Many different algorithms have been studied under a number of different performance measures, and many convergence results have been proven [9].

Theorists have become increasingly interested in the collective behavior of learning automata. Fig. 2 shows collections of N learning automata interacting with an environment. In Fig. 2a, each automaton recieves a different evaluation signal that depends, in general, on the actions of all N automata. This models the situation in which the automata have differing, and possibly conflicting, interests. This a game decision problem. In contrast to the problems studied in classical game theory, the automata operate in total ignorance of the payoff structure of the game and the presence of the other automata. In the case of zero-sum games (games of pure conflict), theoretical results show that when employing certain algorithms, the learning automata converge to the game's solution (by finding the appropriate mixed, or probabilistic, strategies if necessary) [9].

Fig. 2b shows a collection of learning automata in the team situation, which is the special case of the game situation in which the automata receive the same evaluation signal. In this case, the automata have a common goal but each automaton only has partial control over the evaluation. As in the case of games, the learning process in this case is incompletely understood, but a number of mathematical results have been proven, the strongest of which shows that certain stochastic learning automaton algorithms lead to monotonic increases in performance [10].

COMPARISON OF LEARNING AUTOMATA AND NETWORK ELEMENTS

Comparing stochastic learning automata and the typical adaptive elements used in theoretical neural-network research reveals several important differences. First, a typical neuron-like adaptive element has multiple input pathways that carry patterned stimulus information. Such an element might also have a pathway specialized for training, such as the pathway for the desired response of a Widrow/Hoff Adaline [11] or a Perceptron element [12]. The learning process causes the element to implement or approximate a desired mapping from stimulus patterns to responses. A learning automaton, on the other hand, only has a single input pathway for the evaluation signal. Learning either results in the selection of a single optimal action or a suitable action probability vector—no (nontrivial) mapping is produced. On this dimension of comparison, then, the usual adaptive elements are doing something more sophisticated than are learning automata.

However, the usual adaptive element requires an environment that directly provides either a desired response or a signed error that directly tells the element what response it should have produced. In contrast, a learning automaton has to discover, in a stochastic environment, which action is best by sequentially producing actions and observing the results. Since there are no constraints on the success probabilities, information gained from performing one action provides no information about the consequences of the other actions. This can be a nontrivial problem even in the case of two possible actions and is fundamentally different from the supervised learning problem [13,14,15]. Therefore, in terms of the amount of information required for successful learning, a stochastic learning automaton implements a form of learning more powerful than the supervised learning performed by most neuron-like adaptive elements.

Because typical network adaptive elements and learning automata excel on different dimensions, it has been fruitful to study learning elements that combine the capabilities of these two types of systems. The resulting elements are able to learn mappings in the absence of explicit instructional information. In the next section, I describe one algorithm that has resulted from our study of this class of learning elements. Then I discuss the implications of this class of hybrid algorithms for game and team decision problems.

THE ASSOCIATIVE REWARD-PENALTY ELEMENT

The Associative Reward-Penalty element [14], or A_{R-P} element, combines aspects of familiar neuron-like adaptive elements with properties of stochastic learning automata. It is a refinement of similar elements that my colleagues and I have studied [13,16,17,18]. An A_{R-P} element has input pathways $x_1, \ldots, x_n$, which carry pattern input to the element. Each input pathway x_i has an associated weight w_i. An additional input pathway is specialized for delivering environmental evaluation to the element. We call this pathway the reinforcement pathway, r. There is a single output pathway for the element's action, a. Let $\vec{x}(t)$, $\vec{w}(t)$, $r(t)$, and $a(t)$ respectively denote inout pattern vector, weight vector, reinforcement signal, and action at time t. The action is determined by comparing the inner product of the pattern and weight vectors with a randomly varying threshold:

$$a(t) = \begin{cases} 1, & \text{if } \vec{w}(t) \cdot \vec{x}(t) + \eta(t) > 0; \\ 0, & \text{otherwise}; \end{cases} \tag{1}$$

where the $\eta(t)$ are independent identically distributed random variables, each having distribution function Ψ, which is a known and fixed characteristic of the element. Let $p(t)$ denote prob$\{a(t) = 1|\vec{x}(t)\}$. Then

$$p(t) = \text{prob}\{\vec{w}(t) \cdot \vec{x}(t) + \eta(t) > 0\} = 1 - \Psi(-\vec{w}(t) \cdot \vec{x}(t)).$$

Thus, the action probabilities depend on the element's input in a manner parameterized by the weights and the distribution function Ψ. This input-to-probability mapping is adjusted by updating the weight vector according to the following equation:

$$\vec{w}(t+1) - \vec{w}(t) = \begin{cases} \rho[a(t) - p(t)]\vec{x}(t), & \text{if } r(t) = \text{reward};\\ \lambda\rho[1 - a(t) - p(t)]\vec{x}(t), & \text{if } r(t) = \text{penalty}; \end{cases} \tag{2}$$

where $0 \leq \lambda \leq 1$ and $\rho > 0$.

As $|\vec{w}(t) \cdot \vec{x}(t)|$ increases for all $\vec{x}(t)$, the mapping (1) approaches a deterministic linear discriminate function. According to (2), $\vec{w}(t)$ changes in such a way that in the case of reward, the element is more likely to produce the same action, $a(t)$, when patterns similar to $\vec{x}(t)$ occur in the future; in the case of penalty, $\vec{w}(t)$ changes in such a way that the element is more likely to produce the other action, $1 - a(t)$, when patterns similar to $\vec{x}(t)$ occur in the future. Therefore, the A_{R-P} element is an associative learning automaton capable of learning a mapping rather than just a single optimal action. It reduces to a (nonassociative) stochastic learning automaton algorithm when the input pattern is constant and nonzero over time. When it is deterministic ($\eta(t) = 0$ for all t), it becomes the Perceptron algorithm if one treats the terms $a(t)$ and $1 - a(t)$ as the training input giving the desired response. It is most closely related to the "selective bootstrap adaptation" algorithm of Widrow, Gupta, and Maitra [19].

If the input vectors are linearly independent and the distribution function Ψ is continuous and strictly monotonic, one can choose parameters ρ and λ so that the A_{R-P} element converges as closely as desired to the optimal mapping. This holds for arbitrary environmental success probabilities specified by a function $d : X \times A \rightarrow [0, 1]$, where $d(\vec{x}, a) = \text{prob}\{r(t) = \text{success}|\vec{x}(t) = \vec{x}, a(t) = a\}$ (X is the set of input vectors and $A = \{0, 1\}$ is the set of actions). This means that the element will eventually respond to each input vector $\vec{x} \in X$ with the action $a_{\vec{x}}$ with probability as close to 1 as desired, where $a_{\vec{x}}$ is such that $d(\vec{x}, a_{\vec{x}}) = \max\{d(\vec{x}, 0), d(\vec{x}, 1)\}$. Details are provided in ref. 14. Note that if there is a single input pattern, this task reduces to the learning automaton task described above (Fig. 1). When there are many input patterns, the task reduces to a conventional supervised learning pattern classification task with a noisy teacher only if for all $\vec{x} \in X$, $d(\vec{x}, 0) + d(\vec{x}, 1) = 1$. This restriction implies that the evaluation signal provides as much information as a noisy teacher in the supervised learning task [14,15].

TEAM DECISION PROBLEMS WITH ASSOCIATIVE LEARNING AUTOMATA

The simplest team of associative learning automata, such as A_{R-P} elements, consists of N automata sharing the same pattern input and evaluation signal. We have called this system an "Associative Search Network" [16] and have produced several illustrations of its capabilities [13,17,20]. Using the terminology of team decision theory [21], the "information structure" is one in which the team members share the same environmental information but have different points of control over the evaluation. The associative mappings this kind of network can form have exactly the same properties as the mappings formed by the more familiar non-recurrent associative memory networks [22], but they are formed in the absence of explicit instructional information.

Fig. 3 shows a team of associative learning automata with a more complex information structure (the evaluation pathways are not shown). The elements in the leftmost layer have access to environmental state information but have no direct control over the evaluation; the rightmost elements are in the opposite situation; and the interior, or "hidden", elements neither directly sense nor control the team's external environment. However, A_{R-P} elements are able to take advantage of signals provided by other elements in order to improve performance. For example, we set the environmental evaluation criteria so that in order to maximize success

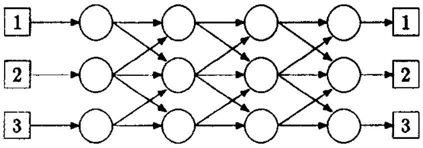

Figure 3. Layered network as a team.

probability the network in Fig. 3 had to implement transmission pathways from input 1 to output 3, input 2 to output 2, and input 3 to output 1. Thus, for a team member to be successful it had to learn to participate in a network that logically formed crossed tranmission pathways through the four layers. A_{R-P} elements solved this task by setting appropriate interconnection weights. This is a form of cooperation by which the elements coordinate their actions in order to increase success probability. As illustrated elsewhere [15], A_{R-P} elements cooperate in a similar manner to form nonlinear mappings if doing so maximizes success probability.

Williams [23] has shown that if all the A_{R-P} elements in an arbitrary acyclic network use the logistic distribution $\Psi(s) = 1/(1+e^{-s})$ for the random variable in (1) and $\lambda = 0$, then the expected change of any weight in the network is proportional to the partial derivative of the overall network's reward probability with respect to that weight. This means that in a probabilistic sense the collective activity of A_{R-P} elements is gradient-following in the appropriate way (although λ has to be nonzero for networks to avoid absorbing at nonoptimal states). In fact, it may not be misleading to regard an A_{R-P} network as a kind of stochastic approximation to a network using the error-backpropagation algorithm recently developed by Rumelhart, Hinton, and Williams [24] in which weights change according to exact gradient information.

Because the weights in A_{R-P} networks only have available an unbiased estimate of the gradient, many more presentations of the input patterns are required than for comparable backpropagation networks but less computation per step is needed. For example, the transmission task just described requires approximately 15,000 to 20,000 steps for solution by an A_{R-P} network, which is probably considerably more than a backpropagation network would require. This assumes, however, that only one action for each element is evaluated for each presentation of an input pattern. Letting the elements perform many actions for each presentation can improve the accuracy of the gradient estimate and reduce the number of presentations required. We do not yet have precise data on the relative amounts of computer time required by these learning methods, but it appears that backpropagation has a substantial edge for conventional sequential simulations [25]. Note, however, that networks of associative learning automata, such as A_{R-P} elements, are applicable to tasks in which error-backpropagation does not apply. These are tasks in which the environment can evaluate the consequences of the network's actions but cannot provide individual error signals for the network's output elements. This type of task arises in learning control problems and is discussed elsewhere [13,26].

CONCLUSION

Cooperativity in neural networks probably takes many forms, some of which are surely represented in the mathematical models and computer simulations to which the label cooperative computation has been applied. The research described in this article is an attempt to add another level of meaning to computational cooperativity by starting with self-interested elements possessing robust adaptive strategies for furthering those interests in relatively unconstrained environments. I have discussed how teams of such elements learn to coordinate their collective behavior in order to solve problems that individual elements cannot solve due to lack of information, control, or representational power. We have not yet investigated more general game decision problems in which the elements have conflicting interests, but the elements' ability to act conditionally upon the decisions of other elements should lead to collective behavior that is more complex than that obtainable from nonassociative learning automata in similar situations.

REFERENCES

1. S. Amari, M. A. Arbib, Competition and Cooperation in Neural Nets (Springer-Verlag, NY, 1982).
2. Feldman, J. A., Ballard, D. H., Cog. Sci. 6, 205 (1982).
3. G. E. Hinton, T. J. Sejnowski, Proc. Fifth Ann. Conf. Cog. Sci. Soc. (1983).
4. D. E. Rumelhart, D. Zipser, In D. E. Rumelhart, J. L. McClelland, eds., Parallel Distributed Processing: Explorations in the Microstructure of Cognition (Bradford Books/MIT Press, Cambridge, MA, to appear).
5. J. J. Hopfield, Proc. Nat. Acad. Sci. 79, 2554 (1982).
6. A. H. Klopf, The Hedonistic Neuron: A Theory of Memory, Learining, and Intelligence (Hemishere, Washington, D.C., 1982).
7. M. L. Tsetlin, Automaton Theory and Modeling of Biological Systems (Academic Press, NY, 1973).
8. H. Robbins, Bull. Amer. Math. Soc. 58, 527 (1952).
9. K. S. Narendra, M. A. L. Thathachar, IEEE Trans. Sys., Man, Cybern. 4, 323 (1974).
10. K. S. Narendra, R. M. Wheeler, IEEE Trans. Sys., Man, Cybern. 13, 1154 (1983).
11. B. Widrow, M. E. Hoff, 1960 WESCON Convention Record Part IV, 96 (1960).
12. F. Rosenblatt, Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms (Spartan Books, Wash., D.C., 1961).
13. A. G. Barto, R. S. Sutton, and C. W. Anderson, IEEE Trans. Sys., Man, Cybern. 13, 846 (1983).
14. A. G. Barto, P. Anandan, IEEE Trans. Sys., Man, Cybern. 15, 360 (1985).
15. A. G. Barto, Human Neurobiology 4, 229 (1985).
16. A. G. Barto, R. S. Sutton, and P. S. Brouwer, IEEE Trans. Sys., Man, Cybern. 40, 201 (1981).
17. A. G. Barto, R. S. Sutton, Biol. Cybern. 42, 1 (1981).
18. R. S. Sutton, PhD Dissertation, University of Massachusetts (1984).
19. B. Widrow, N. K. Gupta, and S. Maitra, IEEE Trans. Sys., Man, Cybern. 5, 455 (1973).
20. O. Selfridge, R. S. Sutton, A. G. Barto, Proc. Ninth IJCAI (1985).
21. Y. C. Ho, Proc. IEEE 68, 644 (1980).
22. T. Kohonen, Associative Memory: A System Theoretic Approach (Springer, Berlin, 1977).
23. R. J. Williams, Technical Report, Institute for Cognitive Science, University of California at San Diego (to appear).
24. D. E. Rumelhart, G. E. Hinton, and R. J. Williams, In D. E. Rumelhart, J. L. McClelland, eds., Parallel Distributed Processing: Explorations in the Microstructure of Cognition (Bradford Books/MIT Press, Cambridge, MA, to appear).
25. C. W. Anderson, PhD Dissertation, University of Massachusetts (to appear).
26. A. G. Barto, Proc. IEEE Workshop on Intelligent Control, Rensselaer Polytechnic Institute, Troy, NY, 1985 (to appear).