Efficient Multi-Agent Reinforcement Learning through Automated Supervision

(Short Paper)

Chongjie Zhang
Computer Science Department
140 Governors Drive
University of Massachusetts
Amherst, MA 01002-9264
chongjie@cs.umass.edu

Sherief Abdallah
Institute of Informatics
British University in Dubai
Knowledge Village, Block 17
Dubai, United Arab Emirates
sherief.abdallah@buid.ac.ae

Victor Lesser
Computer Science Department
140 Governors Drive
University of Massachusetts
Amherst, MA 01002-9264
lesser@cs.umass.edu

ABSTRACT

Multi-Agent Reinforcement Learning (MARL) algorithms suffer from slow convergence and even divergence, especially in large-scale systems. In this work, we develop a supervision framework to speed up the convergence of MARL algorithms in a network of agents. The framework defines an organizational structure for automated supervision and a communication protocol for exchanging information between lower-level agents and higher-level supervising agents. The abstracted states of lower-level agents travel upwards so that higher-level supervising agents generate a broader view of the state of the network. This broader view is used in creating supervisory information which is passed down the hierarchy. We present a generic extension to MARL algorithms that integrates supervisory information into the learning process, guiding agents' exploration of their state-action space.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence

General Terms

Algorithms, Experimentation

Keywords

Reinforcement Learning, Multiagent Systems, Supervision, Heuristics

1. INTRODUCTION

The main contribution of this paper is the development of a framework that speeds up the convergence of Multi-Agent Reinforcement Learning (MARL) algorithms [2, 6] in a network of agents. Each agent's learning occurs in the context of a limited set of agents. We call this set of agents the agent's neighborhood that is specified as an overlay network.

Cite as: Efficient Multi-Agent Reinforcement Learning through Automated Supervision (Short Paper), Chongjie Zhang, Sherief Abdallah, Victor Lesser, *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons(eds.),May,12-16.,2008,Estoril,Portugal,pp. 1365-1368.

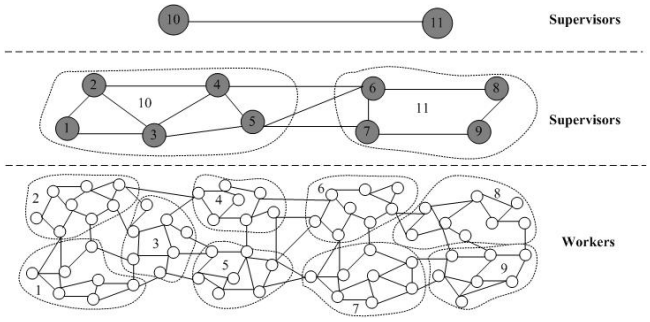

Figure 1: An organization structure for multi-level supervision

The slowness of MARL convergence is due to the large policy search space. Each agent's policy not only includes its local state and actions but also some characteristics of the states and actions of its neighboring agents [2], or the state size of each agent may be proportional to the size of the system [6]. Convergence is also affected by the non-stationarity of the environment (other agents are simultaneously learning their own policies).

Our framework consists of three main components: a multi-level supervision organization (a meta-organization built on top of the agents' overlay network), a communication protocol for exchanging information between lower-level agents and higher-level supervising agents, and a generic extension to MARL algorithms that integrates supervisory information into the learning process. The key idea of our framework is as follows. Each level in the supervising organization is an overlay network in itself. For example, Figure 1 shows a three-level supervision organizational structure. The abstracted states of lower-level agents travel upwards so that higher-level supervising agents can generate a broader view of the state of the network. This broader view comes from not only information about the states of lower-level agents but also information from neighboring supervising agents. In turn, this broader view results in creating supervisory information which is passed down the hierarchy. The supervisory information guides agents in collectively exploring their state-action spaces more efficiently, and consequently results in faster convergence.

2. RELATED WORK

Two paradigms have been studied to speed up the learning process. The first paradigm is to reduce the policy search space. For example, the TPOL-RL [10] reduced the state space by mapping states onto a limited number of action-dependent features. The hierarchical multi-agent reinforcement learning [7] used the explicit task structure to restrict the space of policies, where each agent learned joint abstract action-values by communicating with each other only the state of high-level subtasks. The second paradigm is to use heuristics to guide the policy search. The work described in [11] used both local and global heuristics to accelerate the learning process in a decentralized multirobot system. The local heuristic used only the local information and the global heuristic used the information that was shared and required to be exactly the same among robots. The Heuristically Accelerated Minimax-Q (HAMMQ) [4] incorporated heuristics into the Minimax-Q algorithm to speed up its convergence rate, which shared the convergence property with Minimax-Q. HAMMQ was intended for a two-agent configuration and further the authors had no discussion about how heuristics were constructed.

Our approach follows the second paradigm that uses heuristics to guide the policy search. However, it differs from other approaches in a key respect that it defines a decentralized hierarchical supervision mechanism to automate the generation of heuristics and integrates heuristics into existing unsupervised MARL algorithms (e.g., ReDVaLeR [3], WoLF-GIGA [5], WPL [1], etc.) in a generic manner to speed up their convergence.

3. ORGANIZATIONAL SUPERVISION

Supervision mechanisms commonly exist in human organizations (e.g., enterprises and governments), whose purpose is to run an organization effectively and efficiently to fulfill organizational goals. Supervision involves gathering information, making decisions, and providing directions to regulate and coordinate actions of organization members. The practical effectiveness of the supervision in human organizations, especially in large organizations, inspired us to introduce a similar mechanism into multi-agent systems to improve the efficiency of MARL algorithms.

To add a supervision mechanism to a MAS with an overlay structure, we adopt a multi-level, clustered organizational structure. Agents in the original overlay network, called workers, are clustered based on some measure (e.g., geographical distance). Each cluster is supervised by one agent, called the supervisor, and its member agents are called subordinates. The supervisor role can be played by a dedicated agent or one of the workers. If the number of supervisors is large, higher-level supervisors can be added, and so on, forming a multi-level supervision structure.

Two supervisors at the same level are adjacent if and only if at least one subordinate of one supervisor is adjacent to at least one subordinate of the other. Communication links, which can be physical or logical, exist between adjacent workers, adjacent supervisors, and subordinates and their supervisors. Figure 1 shows a three-level organizational structure. The bottom level is the overlay network of workers which forms 9 clusters. A shaded circle represents a supervisor, which is responsible for a corresponding cluster. Note that links between subordinates and their supervisors are omitted in this figure.

4. COMMUNICATION PROTOCOL

Three types of communication messages, *report*, *suggestion*, and *rule*, are used. A worker's report passes its activity data upwards to provide its supervisor with a broader view. A supervisor's report aggregates the information of reports from its subordinates. A supervisor sends its report to its adjacent supervisors at the same level in addition to its immediate supervisor (if any). The supervisor's view is based on not only the agents that it supervises (directly or indirectly) but also its neighboring supervisors. This peer-supervisor communication allows each supervisor to make rational local decisions when directions from its immediate supervisor are unavailable. To prevent supervisors from being overwhelmed and reduce the communication overhead in the network, the information is summarized (abstracted) in reports. Furthermore, reports are only sent periodically.

Based upon this information, a supervisor employs its expertise, integrates directions from its superordinate supervisor, and provides supervisory information to its subordinates. As in human organizations, rules and suggestions are used to transmit supervisory information. A *rule* is defined as a tuple $\langle c, F \rangle$, where

- c: a condition specifying a set of satisfied states
- F: a set of forbidden actions for states specified by c

A *suggestion* is defined as a tuple $\langle c, A, d \rangle$, where

- c: a condition specifying a set of satisfied states.
- A: a set of actions
- d: the suggestion degree, whose range is $[-1, 1]$.

A suggestion with a negative degree, called a *negative suggestion*, urges a subordinate not to do the specified actions. In contrast, a suggestion with a positive degree, called a *positive suggestion*, encourages a subordinate to do the specified action. The greater the absolute value of the suggestion degree, the stronger the impact of the suggestion on the supervised agent.

Each rule contains a condition specifying states where it can be applied. Subordinates are required to obey rules from their supervisors. Due to their imperativeness, correct rules greatly improve the system efficiency, while incorrect rules can lead to inefficient policies. In contrast, suggestions are used to express a supervisor's preference for subordinates' behavior, which may not be completely correct. Therefore, a subordinate does not rigidly adopt suggestions. The effect of a suggestion on a subordinate's local decision making may vary, depending on its current policy and state. A supervisor will refine or cancel rules and suggestions as new or updated information from its subordinates become available.

A set of rules are in conflict if they forbid all possible actions on some state(s). Two suggestions are in conflict if one is positive and the other is negative and they share some state(s) and action(s). A rule conflicts with a suggestion if a state-action pair is forbidden by the rule but is encouraged by the suggestion. In our supervision mechanism, we assume each supervisor itself is rational and will not generate rules and suggestions that are in conflict. However, in a multi-level supervision structure, a supervisor's local decision may

conflict with its superordinate direction. Rules have higher priority than suggestions. There are several strategies for resolving conflicts between rules or between suggestions, such as always taking its superordinate or local rule, stochastically selecting a rule, or requesting additional information to make a decision. The strategy choice depends on the application domain. Note that it may not always be wise to select the superordinate decision, because, although the superordinate supervisor has a broader view, its decision is based on abstracted information. Our strategy for resolving conflicts picks the most constraining rule and combines suggestions by summing the degrees of the strongest positive suggestion and the strongest negative suggestion.

5. MARL UNDER SUPERVISION

Using MARL, each agent gradually improves its action policy as it interacts with other agents and the environment. A *pure* policy deterministically chooses one action for each state. A *mixed* policy specifies a probability distribution over the available actions for each state. Both can be represented as a function $\pi(s, a)$, which specifies the probability that an agent will execute action a at state s. As argued in [9], mixed policies can work better than pure policies in partially observable environments, if both are limited to act based on the current percept. Due to partial observability, most MARL algorithms are designed to learn mixed policies. The rest of this section shows how MARL algorithms learning mixed policies can take advantage of higher-level information specified by rules and suggestions to speed up convergence.

A typical MARL algorithm contains two components: policy (or action-value function) updating and action choice based on the learned policy. One common method to speed up learning is to supply an agent with additional reward to encourage some particular actions [8]. The use of the special reward affects both policy updating and action choice. In a multi-agent context, special rewards may generate a policy that is undesirable in that they may distract from the main goal, which is supported by the normal reward. In contrast, our approach directly biases the action selection for exploration without changing the policy update process. Hence its effect on the final learned policy is transient (can be turned off at any time), while reward shaping has a permanent effect.

As described previously, a rule explicitly specifies undesirable actions for some states and is used to prune the state-action space. Suggestions, on the other hand, are used to bias agent exploration. The strategy adopted for integrating suggestions into MARL is that the lower the probability of a state-action pair, the greater the effect a positive suggestion has on it and the less the effect a negative suggestion has on it. The underlying idea is intuitive. If the agent's local policy already agrees with the supervisor's suggestions, it is going to change its local policy very little (if at all); otherwise, the agent follows the supervisor's suggestions and make a more significant change to its local policy.

Let R and G be the rule set and suggestion set, respectively, that a worker received and π be its policy. We define $R(s, a) = \{r \in R|$ state s satisfies the condition $r.c$ and $a \in r.F\}$[1] and $G(s, a) = \{g \in G|$ state s satisfies the condition $g.c$ and $a \in g.A\}$. Then a new function π^{AC} for the action choice is defined as:

$$\pi^A(s,a) = \begin{cases} 0 & \text{if } R(s,a) \neq \emptyset \\ \pi(s,a) + \pi(s,a) * \eta(s) & \\ \qquad * \, deg(s,a) & \text{else if } deg(s,a) \leq 0 \\ \pi(s,a) + (1 - \pi(s,a)) & \\ \qquad * \, \eta(s) * deg(s,a) & \text{else if } deg(s,a) > 0 \end{cases}$$

where $deg(s, a)$ and $\eta(s)$ are defined as following.

The function $deg(s, a)$ determines the impact of suggestions. We define $deg(s, a) = max(\{g.d > 0 | g \in G(s, a)\}) + min(\{g.d < 0 | g \in G(s, a)\})$.[2] With this definition, a worker only considers the strongest suggestion, either positive or negative. This definition is also used to resolve conflicting suggestions (in a multi-level supervision organization) by summing the degrees of the strongest positive suggestion and the strongest negative suggestion.

The function $\eta(s)$ is state-dependent and ranges from $[0, 1]$. It determines the receptivity for suggestions and allows the agent to selectively accept suggestions based on its current state. For instance, if an agent becomes more confident in the effectiveness of its local policy on state s because it has more experience with it, then $\eta(s)$ decreases as learning progresses. For example, we set $\eta(s) = k/(k + visits(s))$ where k is a constant and $visits(s)$ returns the number of visits on the state s.

To normalize π^{AC} such that it sums to 1 for each state, the *limit* function from GIGA [13] is applied with minor modifications so that every action is explored with minimum probability ϵ:

$$\pi^{AC} = limit(\pi^{AC}) = argmin_{x:valid(x)} |\pi^{AC} - x|$$

i.e., $limit(\pi^{AC})$ returns a valid policy that is closest to π^{AC}.

We have tested our approach in a distributed task allocation problem. Experimental results show that our approach incorporated with some simple domain knowledge not only dramatically speeds up the convergence rate, but also increases the likelihood of convergence when an unsupervised MARL algorithm fails to converge. Due to the space limit, we describe our experiments in the technical report [12].

6. CONCLUSIONS

This work presents a scalable and robust framework that enables efficient learning in large-scale multi-agent systems. In our framework, the automated supervision mechanism fuses activity information of lower-level agents and generates supervisory information that guides and coordinates agents' learning process. This supervision mechanism continuously interacts with the learning process to accelerate the convergence.

7. REFERENCES

[1] S. Abdallah and V. Lesser. Learning the task allocation game. In *AAMAS'06*, 2006.

[2] S. Abdallah and V. Lesser. Multiagent reinforcement learning and self-organization in a network of agents. In *AAMAS'07*, 2007.

[3] B. Banerjee and J. Peng. Performance bounded reinforcement learning in strategic interactions. In *AAAI'04*, pages 2–7, 2004.

[1] We use "." as a projection operator. For example, $r.c$ returns the rule condition of rule r.

[2] If $G(s, a)$ is empty, then $deg(s, a) = 0$.

[4] R. A. C. Bianchi, C. H. C. Ribeiro, and A. H. R. Costa. Heuristic selection of actions in multiagent reinforcement learning. In *IJCAI'07*, Hyderabad, India, 2007.

[5] M. Bowling. Convergence and no-regret in multiagent learning. In *NIPS'05*, pages 209–216, 2005.

[6] J. A. Boyan and M. L. Littman. Packet routing in dynamically changing networks: A reinforcement learning approach. In *NIPS'94*, volume 6, pages 671–678, 1994.

[7] R. Makar, S. Mahadevan, and M. Ghavamzadeh. Hierarchical multi-agent reinforcement learning. In *Autonomous Agents'01*, pages 246–253, 2001.

[8] A. Y. Ng, D. Harada, and S. Russell. Policy invariance under reward transformations: theory and application to reward shaping. In *ICML'99*, pages 278–287, 1999.

[9] S. P. Singh, T. Jaakkola, M. L. Littman, and C. Szepesvari. Convergence results for single-step on-policy reinforcement-learning algorithms. *Machine Learning*, 38(3):287–308, 2000.

[10] P. Stone and M. Veloso. Team-partitioned, opaque-transition reinforcement learning. In *Autonomous Agents'99*, pages 206–212, 1999.

[11] P. Tangamchit, J. Dolan, and P. Khosla. Learning-based task allocation in decentralized multirobot systems. In *DARS'00*, pages 381–390, 2000.

[12] C. Zhang, S. Abdallah, and V. Lesser. Improving multi-agent learning through automated supervisory policy adaptation. In *University of Massachusetts Amherst Computer Science Technical Report #08-03*, 2008.

[13] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *ICML'03*, pages 928–936, 2003.