Self-organisation in an Agent Network via Learning (Extended Abstract)

Dayong Ye
University of Wollongong
NSW 2522 AU
dy721@uow.edu.au

Minjie Zhang
University of Wollongong
NSW 2522 AU
minjie@uow.edu.au

Danny Sutanto
University of Wollongong
NSW 2522 AU
danny@elec.uow.edu.au

ABSTRACT

In this paper, a decentralised self-organisation mechanism in an agent network is proposed. The aim of this mechanism is to achieve efficient task allocation in the agent network via dynamically altering the structural relations among agents, i.e. changing the underlying network structure. The mechanism enables agents in the network to reason with whom to adapt relations and to learn how to adapt relations by using only local information. The local information is accumulated from agents' historical interactions with others.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent systems

General Terms

Algorithms, Experimentation

Keywords

Self-organisation, Learning, Adaptation

1. INTRODUCTION

To cope with complex tasks, agents are usually organised in a network where an agent interacts only with its immediate neighbours in the network. Self-organisation, which is defined as "*the mechanism or the process enabling the system to change its organisation without explicit external command during its execution time* [3]", can be employed in agent networks to improve the cooperative behaviours of agents. In this paper, our contribution focuses on modification of existing relations between agents to achieve a better allocation of tasks in distributed environments. Towards this end, we propose a self-organisation mechanism via multiagent Q-learning. In contrast to the *K-Adapt* mechanism, proposed by Kota *et al.* [2], our mechanism, called *Learn-Adapt*, is unbiased for both agents which jointly adapt their relation.

2. SELF-ORGANISATION MECHANISM

In our model, an agent network comprises a set of collaborative agents, i.e. $A = \{a_1, ..., a_n\}$, situated in a distributed task allocation environment. The task allocation environment presents a continuous dynamic stream of tasks that have to be performed. Each task, Θ, is composed of a set of subtasks, i.e. $\Theta = \{\theta_1, ..., \theta_m\}$. Each subtask, $\theta_i \in \Theta$, requires a particular resource and a specific amount of computation capacity to fulfill. In addition, each subtask has a relevant benefit paid to the agent which successfully completes the subtask. A subtask θ_i is modeled as a token Δ_i which can be passed in the network to find a suitable agent to complete. Each token consists of not only the information about resource and computation requirement of the corresponding subtask, but also the token traveling path which is composed of those agents that the token has past.

In the agent network, instead of a single type of neighbours, there are three types of neighbours, namely *peer*, *subordinate* and *superior* neighbours, which are constituted by two relations, i.e. *peer-to-peer* and *subordinate-superior* relations. The formal definitions of these two relations are given below.

Definition 1. (*Peer-to-Peer*). A *peer-to-peer relation*, denoted as "$\sim$" ($\sim \subseteq A \times A$), is a *Compatible Relation*, which is reflexive and symmetric, such that $\forall a_i \in A : a_i \sim a_i$ and $\forall a_i, a_j \in A : a_i \sim a_j \Rightarrow a_j \sim a_i$.

Definition 2. (*Subordinate-Superior*). A *subordinate-superior relation*, written as "$\prec$" ($\prec \subseteq A \times A$), is a *Strict Partial Order Relation*, which is irreflexive, asymmetric and transitive, such that $\forall a_i \in A : \neg(a_i \prec a_i)$, $\forall a_i, a_j \in A : a_i \prec a_j \Rightarrow \neg(a_j \prec a_i)$ and $\forall a_i, a_j, a_k \in A : a_i \prec a_j \wedge a_j \prec a_k \Rightarrow a_i \prec a_k$.

In order to change relations between agents, there are seven different atomic actions defined in our model, which are $form\_\sim$, $form\_\prec$, $form\_\succ$, $dissolve\_\sim$, $dissolve\_\prec$, $dissolve\_\succ$ and *no_action*. For example, if agent a_i performs action $form\_\prec$ with agent a_j, a_i will become a *subordinate* of a_j. Obviously, actions $form\_\prec$ and $form\_\succ$ are the reverse action of each other, namely that if a_i performs $form\_\prec$ with a_j, a_j has to take $form\_\succ$ with a_i. The atomic actions can be combined together. The meanings of combination actions can be easily deduced from the meanings of atomic actions. For example, the combination action, $dissolve\_\prec + form\_\sim$, which is taken by a_i with a_j, implies that a_i first dissolves a_j from a_i's *superior* and forms a *peer* relation with a_j. It should be noted that an agent at different time steps might possess different available actions.

The aim of our self-organisation mechanism is to improve the efficiency of task allocation in the agent network via changing the network structure, i.e. changing the relations among agents. Our mechanism is based on the past information of the individual agents. Specifically, agents use the information about the former task allocation processes to

Cite as: Self-organisation in an Agent Network via Learning (Extended Abstract), Dayong Ye, Minjie Zhang and Danny Sutanto, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 1495-1496

evaluate their relations with other agents. We formulate our self-organisation mechanism by using a multiagent Q-learning approach. The reason for choosing the Q-learning approach is that it provides a simple and suitable methodology for representing our mechanism in terms of actions and rewards. Before describing our self-organisation mechanism, we first consider a simple scenario with two agents, a_i and a_j, and three available actions for each agent. The reward matrix of the two agents is displayed in Table 1.

Table 1: Reward Matrix of a_i and a_j

a_i \ a_j	$form\_\prec$	$form\_\sim$	$form\_\succ$
$form\_\succ$	$r_i^{1,1}, r_j^{1,1}$	$r_i^{1,2}, r_j^{1,2}$	$r_i^{1,3}, r_j^{1,3}$
$form\_\sim$	$r_i^{2,1}, r_j^{2,1}$	$r_i^{2,2}, r_j^{2,2}$	$r_i^{2,3}, r_j^{2,3}$
$form\_\prec$	$r_i^{3,1}, r_j^{3,1}$	$r_i^{3,2}, r_j^{3,2}$	$r_i^{3,3}, r_j^{3,3}$

Each cell $(r_i^{x,y}, r_j^{x,y})$ in Table 1 represents the reward received by the row agent (a_i) and the column agent (a_j), respectively, if the row agent a_i plays action x and the column agent a_j plays action y. *Algorithm 1* demonstrates our self-organisation mechanism in pseudocode form. The first

Algorithm 1: self-org. Mechanism according to a_i

1 $Candidates_i \leftarrow a_i$ selects agents in the network;
2 **for each** $a_j \in Candidates$ **do**
3 $\quad Act_i \leftarrow available\_actions(a_i, a_j)$;
4 $\quad Act_j \leftarrow available\_actions(a_i, a_j)$;
5 $\quad$ **for each** $x \in Act_i$, $y \in Act_j$ **do**
6 $\quad\quad$ Initialise Q_{ix} and Q_{jy} arbitrarily;
7 $\quad\quad$ **for** $k = 0$ to a predefined integer **do**;
8 $\quad\quad\quad$ calculate $\pi_{ix}(k)$ and $\pi_{jy}(k)$;
9 $\quad\quad\quad$ $Q_{ix}(k+1) = Q_{ix}(k) +$
$\quad\quad\quad\quad \pi_{ix}(k)\alpha(\sum_y r_i^{x,y}\pi_{jy}(k) - Q_{ix}(k))$;
10 $\quad\quad\quad$ $Q_{jy}(k+1) = Q_{jy}(k) +$
$\quad\quad\quad\quad \pi_{jy}(k)\alpha(\sum_x r_j^{x,y}\pi_{ix}(k) - Q_{ix}(k))$;
11 $\quad\quad$ **end for**
12 $\quad$ **end for**
13 $\quad \langle x_{opti}, y_{opti}\rangle \leftarrow argMax_{match(x,y)}(Q_{ix} + Q_{jy})$;
14 $\quad a_i$, a_j take actions x_{opti} and y_{opti}, respectively;
15 $\quad$ **end if**
16 **end for**

component (Line 1) refers to the reasoning aspect about selecting agents to initiate the self-organisation process, which is described in *Algorithm 2*. After selection, both agents, a_i and a_j, estimate which actions are available at the current state (Lines 3 and 4). Then, a_i and a_j learn the Q-value of each available action, separately (Lines 5-11). In Line 6, the Q-value of each action is initialised arbitrarily. In Line 8, π_{ix} indicates the probability regarding agent a_i taking the action x. To calculate π_{ix}, we employ the ϵ-greedy exploration method devised by Gomes and Kowalczyk [1] shown in Equation 1, where $0 < \epsilon < 1$ is a small positive number.

$$\pi_{ix} = \begin{cases} (1-\epsilon) + (\epsilon/n), \text{if } Q_{ix} \text{ is the highest} \\ \epsilon/n \text{ , otherwise} \end{cases} \quad (1)$$

Furthermore, in Line 9, Q_{ix} is the Q-value of action x taken by agent a_i. In Lines 9 and 10, $0 < \alpha < 1$ is the learning rate.

When finishing learning Q-values, a_i and a_j (Line 13) cooperate to find the optimal actions for both of them, where $match(x, y)$ is a function which is used to test whether the actions x and y that are taken by a_i and a_j, respectively, are matched. An action is only matched by its reverse action. Therefore, a_i and a_j have to cooperate to find the actions, which can be matched together and make the sum of their Q-values become maximum.

Algorithm 2 illustrates the reasoning aspect of each agent for selecting a group of agents to initilise the self-organisation process. Each agent has not only the tokens it currently holds but also all the previous tokens incoming and outgoing through it. Then, each agent uses the local information provided by the tokens to choose candidates. Firstly, from Lines 3 to 5, agent a_i identifies the owner of each token stored in a_i's token list, $Tokens_i$, and counts the number of tokens from each owner. On the one hand, if the number of one owner exceeds a predefined threshold and this owner is not a_i's *peer*, or direct *subordinate* or direct *superior*, this owner will be added into the candidates set (Lines 6-9). This can be explained that if an agent is not a neighbour of agent a_i but often delegated tasks to a_i previously, a_i might want to adapt the relation with this agent. On the other hand, if the number of one owner exceeds another predefined threshold and this owner is a_i's *peer*, or direct *subordinate* or direct *superior*, this owner will be also appended into the candidates set (Lines 10-13). This can be explained that if an agent, which is a neighbour of a_i, delegates very few tasks to a_i, then a_i might also want to alter the relation with this agent.

Algorithm 2: Candidates selection of each agent

1 **for each** $a_i \in A$ **do**
2 $\quad Candidates_i \leftarrow \emptyset$;
3 $\quad$ **for each** $\Delta_k \in tokens_i$ **do**
4 $\quad\quad$ statistics of $\Delta_k.owner$;
5 $\quad$ **end for**
6 $\quad$ **if** $\exists$ # of same $\Delta_k.owner > thre_1$ **and**
7 $\quad\quad \Delta_k.owner \notin Neig_i^{\sim} \vee Neig_i^{\succ} \vee Neig_i^{\prec}$ **then**
8 $\quad\quad\quad Candidates_i \leftarrow Candidates_i \cup \{\Delta_k.owner\}$;
9 $\quad$ **end if**
10 $\quad$ **if** $\exists$ # of same $\Delta_k.owner < thre_2$ **and**
11 $\quad\quad \Delta_k.owner \in Neig_i^{\sim} \vee Neig_i^{\succ} \vee Neig_i^{\prec}$ **then**
12 $\quad\quad\quad Candidates_i \leftarrow Candidates_i \cup \{\Delta_k.owner\}$;
13 $\quad$ **end if**
14 **end for**

3. CONCLUSION

This paper introduces a novel self-organisation mechanism which aims to adapt structural relations among agents in a network to achieve efficient task allocation. By using this mechanism, a pair of agents can independently evaluate each available action and jointly make a decision about taking an action to change their relation.

4. ACKNOWLEDGMENTS

This research is supported by Australian Research Council Linkage Projects (LP0991428) and a URC Research Partnerships Grants Scheme, from the University of Wollongong with the collaboration of TransGrid, Australia.

5. REFERENCES

[1] E. R. Gomes and R. Kowalczyk. Dynamic analysis of multiagent q-learning with e-greedy exploration. In *Proc. of ICML'09*, pages 369–376, Montreal, Canada, Jun. 2009.

[2] R. Kota, N. Gibbins, and N. R. Jennings. Self-organising agent organisations. In *Proc. of AAMAS'09*, pages 797–804, Budapest, Hungary, May 2009.

[3] G. D. M. Serugendo, M.-P. Gleizes, and A. Karageorgos. Self-organization in multi-agent systems. *The Knowledge Engineering Review*, 20(2):165–189, 2005.