THE IMMUNE DISTRIBUTED COMPETITIVE PROBLEM SOLVER USING MHC AND IMMUNE NETWORK

NARUAKI TOMA, SATOSHI ENDO, KOJI YAMADA, HAYAO MIYAGI

Department of Information Engineering, Faculty of Engineering,
University of the Ryukyus, Okinawa 903-0213, Japan
Tel: +81-98-895-8714, Fax: +81-98-895-8727
E-mail: tnal@eva.ie.u-ryukyu.ac.jp, {endo,koji,miyagi}@ie.u-ryukyu.ac.jp

Abstract

The purpose of this paper is to propose an Immune Distributed Competitive Problem Solver using MHC and Immune Network, and to investigate the validity by computer simulations. Our algorithm solves the division-of-labor problems for each agent's work domain in multi-agent system (MAS) by two immune functions. First, Major Histocompatibility Complex (MHC), distinguishes a "self" from the other "nonself", is used to eliminate of competition states. Second, Immune Network which produces specific antibodies by modification of immune cells, is used to produce of adaptive behaviors for agents. Then, to investigate the validity of the proposed method, this algorithm is applied to "N-th agent's Travelling Salesmen Problem (called n-TSP)" as a typical problem of multi-agent system. The good property on solving for MAS will be clarified by some simulations.

Keywords: optimization, MHC and immune network, multi-agent system

1 INTRODUCTION

Such as neural networks and genetic algorithms, adaptive problem solving techniques are based on information processing in biological organisms and are applied many kinds of optimization probelms[1]. On the other hand, the biological immune system is widely recognized as one of the adaptive biological system whose functions are to identify and to eliminate foreign materials. In the biological immune system, following two mechanisms are important to perform the immune functions. First, Major Histocompatibility Complex (MHC) is used to distinguish a "self" from the other "nonself" when nonself invades self. Second, the Immune Network is composed of immune cells and their connections, and specific antibodies are produced by modification of their immune cells.

In this paper, we propose an Immune Distributed Competitive Problem Solver using two immune functions, the MHC and the Immune Network. Our algorithm solves the division-of-labor problems for each agent's work domain in multi-agent system (MAS). The MHC is used to eliminate of competition-states among agents. The Immune Network is used to produce of adaptive behaviors for agents. By introducing such models, we construct an adaptive optimization algorithm to solve division-of-labor problems. Then, we apply the proposed algorithm to n-th agent's travelling salesman problem (called n-TSP) which is typical problem in MAS. Some computer simulations are designed to clarify both a basic performance and characteristics featuers of proposed immune algorithm.

2 MAS AND DIVISION-OF-LABOR PROBLEMS

Multi-agent system which is a study in a field of distributed artificial intelligence is an information processing technique that autonomic agents solves problems by interactions among their agents. Division-of-labor problems are subject focuseed on distribution of works for agents. The solvers of the problems are defined as follows.

[Aims of distributed problem solving]
The domain that each agent covers is defined as work domains (WD), and the total of the work domains is defined as problem domain (PD). There are two aimes, (a) each WD_i that is assigned by $agent_i$ must to be divided evenly and, (b) the system performs effective division-of-labor by optimizing each WD_i.

3 PROPOSED METHOD

3.1 The Immune Distributed Competitive Problem Solver

In our method, transmission of the MHC on the Immune Network and re-construction of the components produce specific antibodies. The MHC is an information carrier that can keep up informations both of self and nonself by changing itself into MHC+peptide. We call the states, which the MHC has informations of nonself, competition states. The details of our method are described as follows.

[Step1. Definition of antigens and MHC]
Problem space that agents exist is defined as antigen, and internal state (e.g. solution, behavior, etc.) of each agent is defined as MHC.

[Step2. Preparation of initial MHC and immune cells]
Prepare all of the initial MHC and immune cells (Macrophage, T cell and B cell) at random.

[Step3. Acquisition of informations]
Macrophage acquires informations both of external and internal one. The external informations are described as MHC with internal information, and the MHC is transmitted to T cell.

[Step4. Elimination of Competition states]
If the competition states exist in the transmitted MHC, T cell eliminate the states by competition processing (the details is described next section).

[Step5. Production of candidate solutions]
On the basis of MHC that consists of the external informations and the internal informations that DOSE NOT include competition states, B cell N produces candidate (similar) solutions that is approximated to the MHC. The design of the similarity and the way of production of similar solutions depend on the problem.

[Step6. Extraction of a superior solution]
Each agent extracts a superior solution from the above similar solutions, and performs the solution. Then, if the terminal condition isn't satisfied, go to step 3.

Four steps, from 3rd to 6th steps, are one time step. This algorithm is adapted by repeating from step 3 to step 6 until terminal condition.

3.2 Competition processing

The competition processing, is performed at Step4 in above algorithm, is a mechanism that eliminates competition states by comparing fitness of between competition agent. The definitions of terms that is used here to explain this mechanism is described to table 1. According to the terms, a competition state is described equation (1).

$$MHC(A) \cap MHC(B) \neq \emptyset \tag{1}$$

	self agent: A	nonself agent: B
work space, now	$MHC(A)$	$MHC(B)$
work space, after	$MHC(A')$	$MHC(B')$
fitness, now	$fitness(A)$	$fitness(B)$
fitness, after	$fitness(A')$	$fitness(B')$

Table 1: Definitions of terms. Consider case of competition states between agent A and B exist (formula (1)). Here 'after' states means that the time after was eliminated the competition. (ie. the states of A' is different from the states of A.)

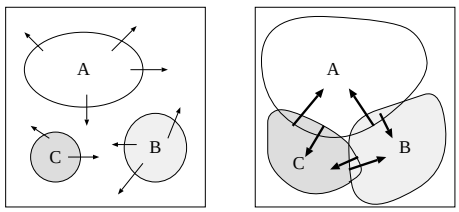
[Step1. Comparison of fitness in self agent]
In the competition processing, it is compared between $fitness(A)$ and $fitness(A')$. If the $fitness(A')$ without competition states is high, the self agent modifies the fitness from $fitness(A)$ to $fitness(A')$ and the MHC from $MHC(A)$ to $MHC(A')$ by loseing competition states to agent B. Otherwise go to Step2.

[Step2. Comparison of fitness between self and nonself agent]
It is compared between $f(A)$ and $f(B)$. If the $f(A)$ is large, the competition states are eliminated from agent B. Otherwise, the competition states are eliminated from agent A.

3.3 Concept of behaviors and aspects

The proposed method consists of (1) elimination of competition using MHC and (2) production of behaviors using Immune Network. On basis of the construction, this method has antithetic adaptations (see also figure 1). According to these interactions, our method can perform to solve the division-of-labor problems efficiency.

1. Each agent adapts in direction of raising the fitness on the inside itself by the production using Immune Network (see left-hand side of figure 1).
2. Each agent adapts in direction of elimination of inefficiency region in each work domain by the elimination of competition (see right-hand side of figure 1).



(A) Amplification of work spaces (B) Abridgement of work spaces

Figure 1: Basic concept of the proposed method.

4 EXPERIMENT

Two computer simulations are performed to investigate the performance of our method. In first simulation, we apply our method to n-th agent's TSP, then consider the results. In second simulation, it is applied Genetic Algorithm (GA) to the same problem, then both performances are compared.

4.1 N-th agent's TSP

We apply to n-th agent's Travelling Salesman Problem to investigate an adaptation ability of our model. Travelling salesman problem (TSP) is one of the most typical combinatorial optimization problems. The object is finding a roundtrip of minimal total length visiting each node exactly once. N-th agent's travelling salesman problem (n-TSP) is extended the definition of the number of salesman from one to the plural number. The object is finding the minimal tour by division among salesman. Note that, two conditions are set. First, the number of salesman is fixed while the system solves. Second, start cities whose all salesmen are corresponded.

4.2 Experiment 1. Definition of simulation

To confirm the basic performance of our method, we apply the method to simple n-TSP. The problem defined in the table 2. The parameters of proposed method are set in figure 2-(A). The $fitness_i$, the fitness of agent i, is defined as equation 2.

$$fitness_i = Cities(MHC(i))/Cost(MHC(i)) + 1/Cost(MHC(i)) \quad (2)$$

- $MHC(i)$: the tour of agent i.
- $Cities(MHC(i))$: the number of cities in the $MHC(i)$.
- $Cost(MHC(i))$: the length of the $MHC(i)$.

the number of cities	25
position of cities	circularity
start city	center of circle
radius	0.4
the number of salesmen	3
time step	100

Table 2: Parameters of n-TSP.

4.3 Experiment 1. Results and discussions

Figure 2-(B) shows an acquired solution for the n-TSP, and is clear that (a) the number of cities (without start city) visited by each agent are 8 and the even division of work spaces is performed, and (b) the solution of each agent is optimum solution.

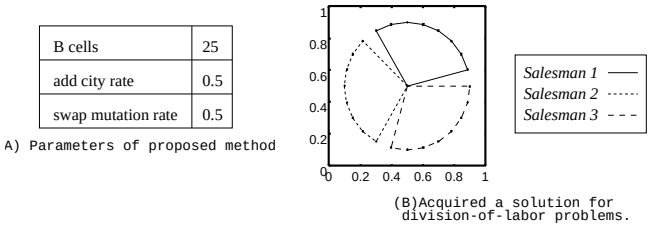


Figure 2: Parameters and result.

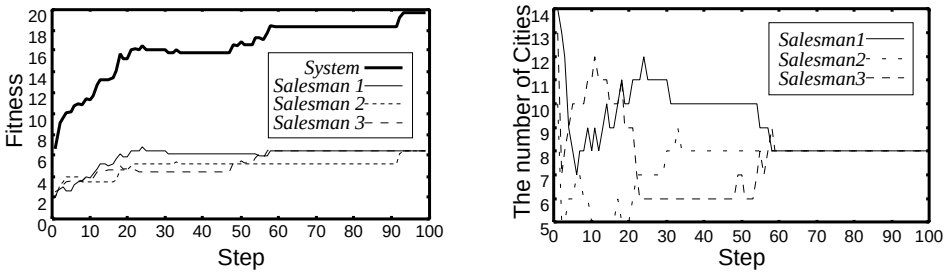


Figure 3: Results, transition of fitness and the number of visited cities, in case of our method.

Next, to verify the behaviors of the method, the transition of the fitness and the number of visited cities shows figure 3.

In figure 3-(A), *System* is the total fitness of all agents. Figure 3-(A) shows that the fitness of *System* transits monotonous increase as a whole though one or more decreases. In our method, re-construction of a candidate solution is executed at following 3 cases: (1) self agent eliminates own competition states, (2) own competition states are eliminated by nonself agent, and (3) own fitness increases by production of similar solutions. In case of (1) and (3), the fitness of self agent increases and otherwise decreases. To clarify more the behaviors of our method, the transition of the number of visited cities for each agent shows figure 3-(B).

In figure 3-(B), the large movements occur until step 30, and the work domain (the number of visited cities) of each agent converges at 8 cities after step 60. It is considered that the work domain of each agent is moved with keeping the fitness because of the fitness is increasing stable (see figure 3-(A)) though their work domains are fluctuated widely. This property will be good for solving division-of-labor problems.

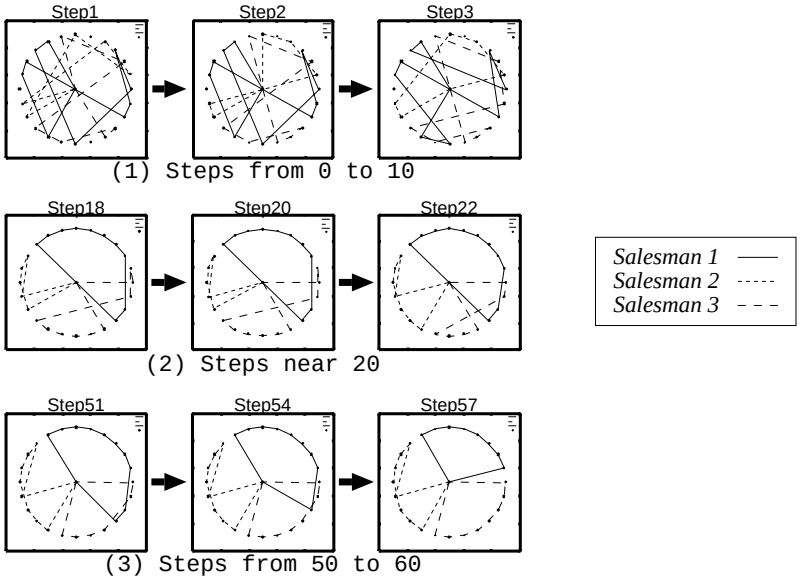


Figure 4: Changing process of solutions for each agent.

Figure 4 shows the changing of solutions for each agent at steps that the movements of their work spaces are large especially. In the steps from 0 to 10, the initial work spaces have a lot of not-used spaces or overlaps therefore re-construction of solutions is executed frequently. In the steps near 20, according to figure 3-(A), the fitness of *System* increases comparatively. And the large movements of work spaces are caused by the decrement of the work domain for *Salesman 3*. The re-construction for applicable solutions is confirmed in figure 4. In the steps from 50 to 60, the decrement of the work domain for *Salesman 1* occurs, and the best solutions for *Salesman 1* and *Salesman 2* are confirmed in figure 4. Finally, the best solutions (see figure 2-(B)) are acquired by optimizing the work domains of *Salesman 2*. It is considered that the work domain of each agent is moved with keeping the fitness because of the fitness is increasing stable (see figure 3-(A)) though their work domains are fluctuated widely. This property will be good for solving division-of-labor problems.

4.4 Experiment 2. Definition of simulation

We compare with the results both our method and GA. The parameters of GA are set in figure 5-(A). The $fitness_i$, the fitness of agent i, is defined as equation 3.

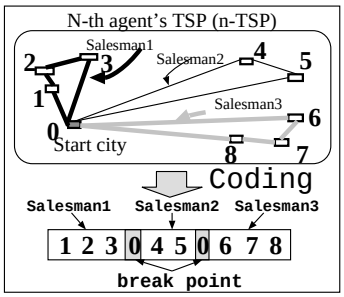
The candidate solutions must be encoded to apply GA to n-TSP. We adopted the pass representation method originally proposed by Yamamura[6]. In subtour crossover, two subtours which have the same elements are looked up, and those subtours are replaced. It becomes possible that adopting thie method prevents the destruction of the important subtour. Coding and genetic operator are shown in the figure 5.

$$fitness_i = 1/(Cost * Cost_W + Penalty * Penalty_W) \tag{3}$$

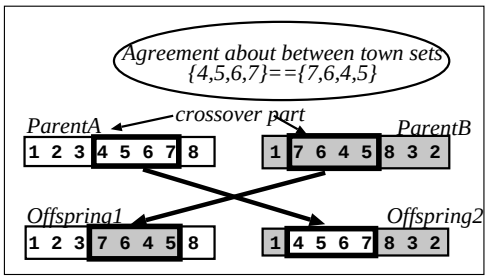
- $Cost$: length of the tour.
- $Cost_W$: weight of Cost.
- $Penalty$: penalty for the equality.
- $Penalty_W$: weight of penalty.

population number	100
crossover rate	1.0
mutation rate	0.01
terminal generations	1000

(A) Parameter of GA.



(B) An example of coding.

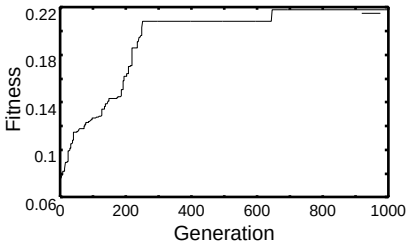


(C) An example of subtour crossover.

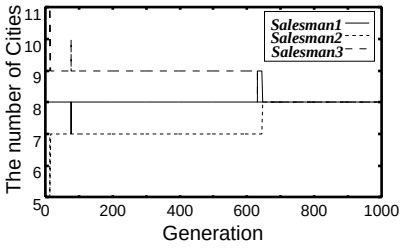
Figure 5: Paremeters, coding and subtour crossover.

4.5 Experiment 2. Results and discussions

Figure 6-(A) shows that the maximum fitness of GA transits monotonous increase, like our method. Note that, the numbers of searched points in two methods are greatly difference. In fact, the number is $25 \times 3 \times 100 = 7,500$ in case of out method. In case of GA, the number of $100 \times 1,000 = 100,000$. Thus, a cost of our method is less than "GA's cost $\times$ 1/10". The reasons of the cost of GA is so large are as follows. In our method, each agent can search independently owns a part of a problem space in parallel. On the other hand, the GA has no other search whole of a problem space at a time. Actually, figure 6-(B) shows that the movement of the number of visited cities is little confirmed. In normal cases using GA, if subtour crossover is performed, the number of visited cities for each salesman is changed. But the changes are little. It means that a genetic operator which performs effectively for evolution will be mutation. The crossover, subtour crossover, cannot actively clear away a obstructive part of problem space, though it is very useful operator to maintain good schemata. In conclusion, it must be mounted the property function, actively clear away an obstructive part of problem space, to solve effectively division-of-labor problems in MAS.



(A) Transition of maximum fitness.



(B) Transition of the number of visited cities for each agent.

Figure 6: Results, transition of maximum fitness and the number of visited cities, in case of GA.

5 CONCLUSION

We proposed models of immune functions and the Immune Distributed Competitive Problem Solver using their models. Our method consists of (1) elimination of competition using MHC and (2) production of behaviors using Immune Network. This method was applied to simple n-TSP, and validity was examined. In the experiments, the property that the work domain of each agent is moved with keeping the fitness was confirmed. And, it is investigated that our method is good performance than the GA. As future works, we're going to extend our method (e.g. extending the number of activated cells from one to plural number, introducing two situations for cell interaction.) and compare with other methods.

ACKNOWLEDGEMENTS

We would like to appreciate Prof. Y. Itoh (Lymphocyte Biology Section, Laboratory of Immunology, National Institute of Allergy and Infectious Diseases, National Institutes of Health, Bethesda MD 20892) and members of Laboratory of Harmonious Systems Engineering, Research Group of Complex Systems Engineering, Division of Systems and Information Engineering, Graduate School of Engineering, Hokkaido University.

The first author acknowledges the Grant-in-Aid for Scientific Research (Grant-in-Aid for JSPS Fellows).

REFERENCES

[1] Thomas Back (1997) Proceedings of The Seventh International Conference on Genetic Algorithms. Michigan State University, East Lansing, MI.

[2] Charles A. Janeway, Jr., Paul Travers ; with assistance of Simon Hunt, Mark Walport (1997) Immunobiology : The Immune System in Health And Disease. Garland Pub.

[3] Y. Ishida, H. Hirayama, H. Fujita, A. Ishiguro, K. Mori. (1998) Immunity-Based Systems and Its Applications. CORONA.

[4] D.E. Goldberg (1989) Genetic algorithm, search optimization and machine learning. Addison Wesley.

[5] S. Forrest, A.A. Perelson (1990) Genetic algorithm and the Immune system. Proc. of 1st Workshop on Parallel Problem Solving from Nature, Dortmund, Federal Republic of Germany, 1-3, October, 1990.

[6] M. Yamamura, T. Ono, S. Kobayashi (1992) Character-Preserving genetic algorithms for Travering salesman problem. Jurnal of JSAI, Vol.7 No.6, pp.1049-1059, 1992.