Applied Soft Computing 2 (2002) 39–47

Applied Soft Computing

www.elsevier.com/locate/asoc

An immunity-based ant colony optimization algorithm for solving weapon–target assignment problem

Zne-Jung Lee [a,*], Chou-Yuan Lee [b], Shun-Feng Su [c]

[a] *Department of Information Management, Kang-Ning Junior College of Nursing, National Taiwan University of Science and Technology, Taipei, Taiwan*
[b] *Department of Information Management, Lan-Yang Institute of Technology, National Taiwan University of Science and Technology, Taipei, Taiwan*
[c] *Department of Electrical Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan*

Abstract

In this paper, an immunity-based ant colony optimization (ACO) algorithm for solving weapon–target assignment (WTA) problems is proposed. The WTA problem, known as a NP-complete problem, is to find a proper assignment of weapons to targets with the objective of minimizing the expected damage of own-force assets. The general idea of the proposed algorithm is to combine the advantages of ACO, the ability to cooperatively explore the search space and to avoid premature convergence, and that of immune system (IS), the ability to quickly find good solutions within a small region of the search space. From our simulation for those WTA problems, the proposed algorithm indeed is very efficient.
© 2002 Published by Elsevier Science B.V.

Keywords: Ant colony optimization; Optimization; Immune system; Weapon–target assignment

1. Introduction

A weapon–target assignment (WTA) problem is to find a proper assignment of weapons to targets with the objective of minimizing the expected damage of own-force assets. WTA in fact is an NP-complete problem, and various methods for solving such NP-complete optimization problems have been reported in the literature [1,9,10]. Classical methods are based on graph search approaches and usually result in exponential computational complexities [10–14]. Thus, when the problem size is large, those methods may need lots of time to find the optimal solutions or sometimes even are not able to find the optimal solutions. Other methods [3], such as simulated annealing (SA) [2,21] or genetic algorithms (GAs) are also widely employed to solve optimization problems and have demonstrated satisfactory performances in various applications. SA has been shown to have the ability of finding the global optimum. However, due to its sequential search characteristics, SA cannot be used in a parallel architecture to improve its search efficiency. GAs can be viewed as parallel search techniques that stimulate the evolution of individual structures for optimization inspired by natural evolution. However, the parallelism of search is based on the solution (or chromosome) level. Thus, the search efficiency may not be very nice. This phenomenon can be seen in our simulations.

Recently, many research activities have been devoted to ant colony optimization (ACO) and immune system (IS) [16,18,19,28,30]. ACO and IS have applied to many fields and shown premising results in various applications [7,8,29,30]. ACO was initially

* Corresponding author.

proposed to solve travel salesman problems (TSP) [26,27]. It is a class of algorithms using artificial ants with the capability of mimicking the behavior of real ants [4–6]. Ants are capable of exploring and exploiting pheromone information, which have been left on the traversed ground. Ants then can choose paths based on the amount of pheromone. With such concept, a multi-agent algorithm called ACO has been widely employed as a cooperative search algorithm for solving optimization problems [7,8]. Because the search parallelism of ACO is based on the components of a solution (or gene) level, the search efficiency of ACO is much better than that of GAs. This claim can be verified in our simulations.

The natural immune system is a very complex system with several mechanisms to defense against pathogenic organisms. However, the natural immune system is also a source of inspiration for solving optimization problems. From the information processing perspective, immune system is a remarkable adaptive system and can provide several important aspects in the field of computation [28,29]. When incorporated with evolutionary algorithms, immune system can improve the search ability during the evolutionary process [36]. In our research, a specific designed immune system is implemented to improve the local search efficiency of ACO. The proposed algorithm indeed demonstrates excellent performance in later simulations.

The paper is organized as follows. In Section 2, the WTA problems is defined and introduced. The idea and algorithm of ACO are described in Section 3. Since local search process is conducted to improve the generated solutions in ACO, the concept of local search and SA are discussed in Section 4. In Section 5, an immunity-based ant colony algorithm is presented and discussed. The results of employing the proposed algorithm to solve WTA problem are presented in Section 6. Several existing search algorithms, ACO, SA, and GAs, were also employed for comparison. The comparisons showed the superiority of the proposed algorithm. Finally, Section 7 concludes the paper.

2. The WTA problem

Usually, it is not an easy task for field planners to make a proper WTA in front of various threat targets. Thus, a decision-aided system of WTA problem is strongly desired in helping and training planners to make proper decisions on the battlefield. In our study, we consider that there are W weapons and T targets. Two assumptions are made for the formulation of WTA problems. The first one is that all weapons must be assigned to targets. The second one is that the individual probability of killing (K_{ij}) by assigning the ith target to the jth weapon is known for all i and j. This probability defines the effectiveness of the jth weapon to destroy the ith target. The overall probability of killing (PK) value for a target (i) to damage the asset is computed as

$$\mathrm{PK}(i) = 1 - \prod_{j=1}^{W} (1 - K_{ij})^{X_{ij}}, \tag{1}$$

where X_{ij} is a Boolean value indicating whether the jth weapon is assigned to the ith target. $X_{ij} = 1$ indicates that the jth weapon is assigned to the ith target. The considered WTA problem is to minimize the following fitness function [1,25]:

$$F(\pi) = \sum_{i=1}^{T} \mathrm{EDV}(i) \times \mathrm{PK}(i), \tag{2}$$

subject to the assumption that all weapons must be assigned to targets, i.e.

$$\sum_{i=1}^{T} X_{ij} = 1, \quad j = 1, 2, \dots, W. \tag{3}$$

Here, EDV(i) is the expected damage value of the ith target to the asset, π is a feasible ETA list, and $\pi(j) = i$ indicates weapon j is assigned to target i.

3. Ant colony optimization (ACO) algorithm

ACO algorithm inspired by colonies of real ants has been successfully employed to solve various optimization problems [15–20]. The general ACO algorithm is illustrated as [23,24] follows.

```
Procedure: ACO algorithm
  Begin
    While (ACO has not been stopped) do
      Schedule activities
        Ant's generation and activity;
```

```
            Pheromone evaporation;
            Daemon actions;
        End schedule activities
    End;
End;
```

The procedure of the ACO algorithm manages the scheduling of three activities: ants' generation and activity, pheromone evaporation, and daemon actions. One commonly used ACO algorithm is the ant colony system (ACS) [41]. In the ant's generation and activities of ACS, the *k*th ant at time *t* positioned on node *r* move to the next node *s* with the rule governed by

$$s = \begin{cases} \arg\{\max_{u=\text{allowed}_k(t)}[\tau_{ru}(t)\eta_{ru}^{\beta}]\}, & \text{when } q \le q_0 \\ S, & \text{otherwise} \end{cases} \tag{4}$$

where $\tau_{ru}(t)$ is the pheromone trail at time *t*, η_{ru} the problem-specific heuristic information, and β is a parameter representing the importance of heuristic information, *q* is a random number uniformly distributed in [0, 1], q_0 is a pre-specified parameter $(0 \le q_0 \le 1)$, allowed$_k(t)$ is the set of feasible nodes currently not assigned by ant *k* at time *t*, and *S* is an index of node selected from allowed$_k(t)$ according to the probability distribution given by

$$P_{rs}^{k}(t) = \begin{cases} \dfrac{\tau_{rs}(t)\eta_{rs}^{\beta}}{\sum_{u\in\text{allowed}_k(t)}\tau_{ru}(t)\eta_{ru}^{\beta}}, & \text{if } s \in \text{allowed}_k(t) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

In finding feasible solutions, ants perform online step-by-step pheromone updates as

$$\tau_{ij}(t+1) \leftarrow (1-\psi)\tau_{ij}(t) + \psi\tau_0 \tag{6}$$

where $0 < \psi \le 1$ is a constant, $\tau_0 = (mF_{nn})^{-1}$ is the initial value of pheromone trails, *m* the number of ants, and F_{nn} is the fitness value produced by the nearest neighbor heuristic [35]. In Eq. (6), pheromone update rule has the effects of making the visited paths less and less attractive if ants do not visit those paths again. It is the pheromone evaporation process. Finally, the daemon actions are performed to update pheromone trails. It activates a local search process to improve the solutions generated by ants and performs offline pheromone trail update. When the best solution has been found, the pheromone trail update rule is performed as

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \rho\,\Delta\tau_{ij}(t) \tag{7}$$

where $0 < \rho \le 1$ is a parameter governing the pheromone decay process, $\Delta\tau_{ij}(t) = 1/F^{\text{elitist}}$, and F^{elitist} is the elitist fitness value from the beginning of the search process.

4. The concept of local search and simulated annealing

In the ACS algorithm, a local search process is also conducted to improve the generated solutions. The local search process is to find better solutions in a local manner. In fact, local search has often been employed in various search algorithms to improve the search efficiency. Local search can explore the neighborhood in an attempt to enhance the fitness value of the solution in a local manner. The procedure of general local search process is [32,33] given as follows.

```
Procedure: general local search
  Begin
    While (General local search has not been stopped) do
      Generate a neighborhood solution $\pi'$
      If $F(\pi') < F(\pi)$ then $\pi = \pi'$
    End;
  End;
```

General local search starts from the current solution and repeatedly tries to improve the current solution by local changes. Traditionally, local changes are to randomly. If a better solution is found, then it replaces the current solution and the algorithm searches from the new solution again. These steps are repeated until a criterion is satisfied. The used criterion usually is to perform M_g times of local changes, where M_g ($<W$) is randomly generated.

Simulated annealing (SA) algorithm makes use of search strategies where cost-deteriorating neighborhood solutions may possibly be accepted as candidates in the search process [37]. In SA, in addition to accepting better-fitness neighbors, worse-fitness neighbors may also be accepted depending on a probability that gradually decreases in the process. SA enables asymptotic convergence to the optimal solution and has been widely used for solving optimization problems [21,22]. The pseudo code of the SA algorithm is described as follows [38,39].

```
Procedure: SA algorithm
  Begin
    Define the initial temperature $T_1$ and the coefficient $\gamma$ $(0 < \gamma < 1)$;
    Randomly generate an initial solution as the current state;
    $\lambda \leftarrow 1$;
    While (SA has not been frozen) do
      $\sigma \leftarrow 0$; $\phi \leftarrow 0$;
      While (equilibrium is not approached sufficiently closely) do
        Generate new solution from the current solution;
        $\Delta F$ = fitness value of new solution − fitness value of current solution;
        $P_r = \exp(-\Delta F/T_\lambda)$;
        If $P_r \geq$ random[0, 1] then
          Accept new solution; current solution ← solution;
          $\phi \leftarrow \phi + 1$;
        End;
        $\sigma \leftarrow \sigma + 1$;
      End;
      Update the maximum and minimum fitness values;
      $T_{\lambda+1} \leftarrow T_\lambda * \gamma$;
      $\lambda \leftarrow \lambda + 1$
    End;
  End;
```

In our implementation, the initial temperature is set as [36]

$$T_{\lambda=1} = \ln\left(\frac{F^{\text{elitist}}}{\lambda+1}\right). \qquad (8)$$

The way of generating new solutions is to inverse two randomly selected positions in the current solution. In this algorithm, the new generated solution is regarded as the next solution only when $\exp(-\Delta F/T_\lambda) \geq$ random[0, 1], where random[0, 1] is a random value generated from a uniform distribution in the interval [0, 1]. It is easy to see that when the fitness value of generated solution is better than that of current solution, ΔF is negative and $\exp(-\Delta F/T_\lambda)$ is always greater than 1. Thus, the solution is always updated. When the new solution is not better than its ancestor, the solution may still take place of its ancestor in a random manner. Such a process is repeated until an equilibrium state is approached sufficiently closely. The equilibrium state is defined as [40,42]

$$\left\{\begin{matrix} \sigma \geq \Gamma, & \text{or } \phi \geq \Phi \\ \Gamma = 1.5W, & \Phi = W \end{matrix}\right\} \qquad (9)$$

where σ is the number of new solutions generated, ϕ the number of new solutions accepted, Γ the maximum number of generations, and Φ is the maximum number of accepted solutions. This algorithm is repeated until it reaches a situation, which is:

$$\frac{F^{\max} - F^{\min}}{F^{\max}} \leq \varepsilon \quad \text{or} \quad T_\lambda \leq \varepsilon_1, \qquad (10)$$

where $F^{\max}$ and $F^{\min}$ are the maximum and minimum fitness values, and ε and ε_1 are pre-specified constants ($\varepsilon = 0.001$, $\varepsilon_1 = 0.005$ in our implementation).

5. Immune system and the proposed algorithm

Recently, the immune system provides an alternative in which problem-specific heuristics can be embedded into search process. Biologically, the function of an immune system is to protect our body from antigens. Several types of immunity, such as anti-infected immunity, self-immunity, and particularity immunity are investigated in biology. In the aspect of self-immunity, there are many kinds of antibodies against self-antigen in the body. They are helpful in eliminating the decrepit and degenerative parts but will not destroy the normal parts [29,31,36]. From the concept of self-immunity, a novel model for the immune system has been developed in [36] to embed heuristics for genetic algorithms for solving travel salesman problem. The approach consists of two main features. The first is vaccination used for reducing the current fitness value. It uses heuristics to repair bits in current solutions, and possibly solutions of better fitness value are obtained. The other is immune selection used for preventing deterioration. It includes two steps. The first one is called the immune test, and the second one is the annealing selection. The immune test is used to test the vaccinated bits, and the annealing selection is to accept these bits with a probability according to their fitness value. The detailed description of those approaches can be found in [36]. In our implementation, we use the similar ideas to embed heuristics into ACO.

In this paper, an immunity-based ACO algorithm is proposed to enhance the search performance for solving WTA problem. It combines the advantages that ACO, a colony of artificial ants, cooperates together to find better solutions and performs global search to escape from local optima, and immune system utilizes problem-specific heuristics to conduct local search and fine-tuning in the solution spaces. The diagram of the proposed algorithm is shown in Fig. 1. In the proposed

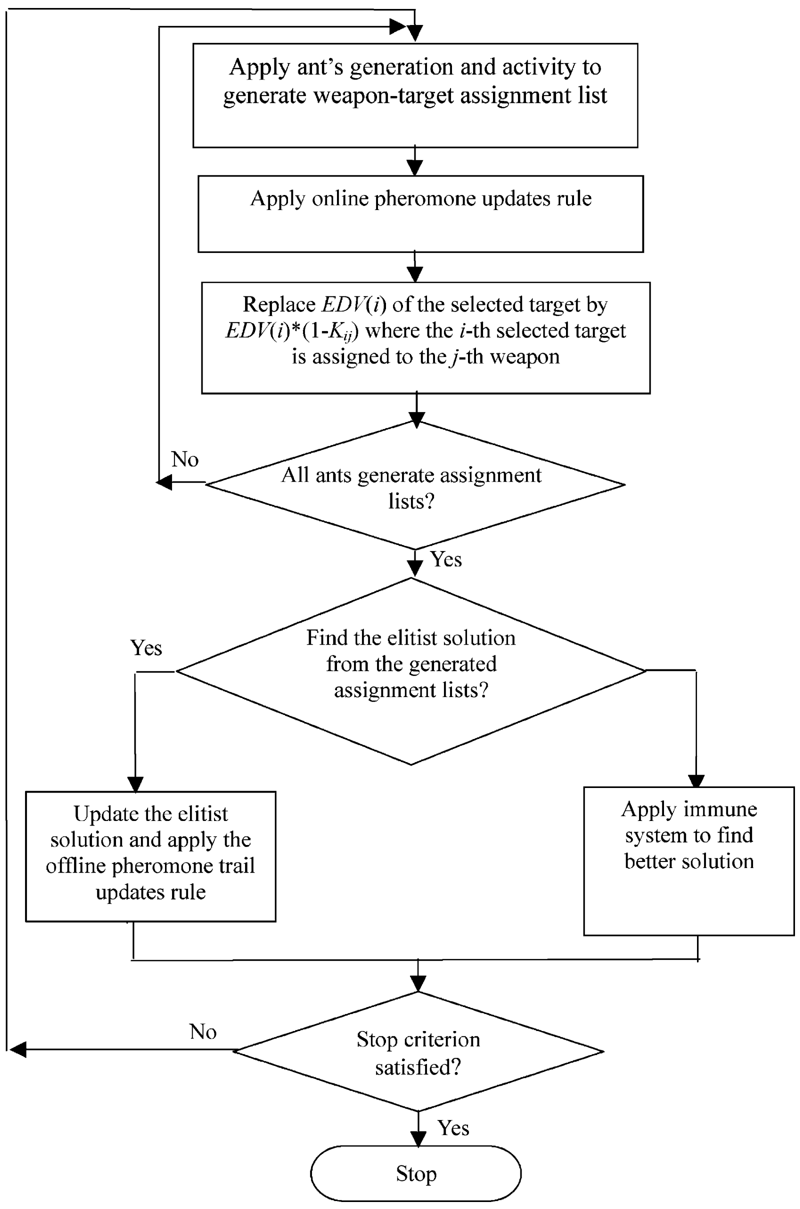
Apply ant's generation and activity to generate weapon-target assignment list
Apply online pheromone updates rule
Replace *EDV*(*i*) of the selected target by *EDV*(*i*)*(1-*K*ij) where the *i*-th selected target is assigned to the *j*-th weapon
No
All ants generate assignment lists?
Yes
Yes
Find the elitist solution from the generated assignment lists?
Update the elitist solution and apply the offline pheromone trail updates rule
Apply immune system to find better solution
No
Stop criterion satisfied?
Yes
Stop

Fig. 1. The diagram of the proposed algorithm.

algorithm, like that in traditional ACO, ants' generation and activity is first applied to generate feasible WTA lists. In this process, each ant chooses a weapon randomly and successively assigns targets to weapons until weapons have been completely assigned. The rule for ant k assigning target i to weapon j is governed by

$$\pi(j)=\begin{cases}\arg\{\max_{i=\mathrm{allowed}_k(t)}[\tau_{ij}(t)\eta_{ij}^{\beta}]\}, & \text{when } q \le q_0\\ S, & \text{otherwise}\end{cases} \tag{11}$$

The heuristic information (η_{ij}) is set as the highest value of $K_{ij} \times \mathrm{EDV}(i)$. Thereafter, the online pheromone update rule is applied according to Eq. (6), and EDV(i) of the selected target is changed to $\mathrm{EDV}(i) \times (1 - K_{ij})$ because the ith selected target has already been assigned to the jth weapon. After generating all assignment lists, if a new elitist solution is found then the pheromone trail update rule of Eq. (7) is applied. On the other hand, immune system is conducted to find a better solution. The diagram of immune system is shown in Fig. 2. First, the best assignment list is chosen from all assignment lists. Then vaccination and immune selection are applied to the best assignment list then to find a better solution. The better the assignment pairs exist in an assignment list, the more likely it is an optimal assignment. Thereafter, vaccination is to repair bits and reduce fitness value in the current solution. In our implementation, the nice bits are kept, and other bits are repaired to find possible better assignment pair in the solution. The jth bit is said to be nice enough if $K_{ij} \times \mathrm{EDV}(i)$ of its corresponding assignment of the jth weapon to the target is the highest among all i. When a bit is not nice enough, it will be repaired by a randomized integer between 1 to T. In the immune test, the repaired bits with better fitness values are accepted, and those of repaired bits with worse fitness values may be also accepted according to the annealing selection. The selected probability for the jth repaired bit in annealing selection is calculated as

$$P^j = \frac{e^{-F_\lambda^{ij}/T_\lambda}}{\sum_{j=1}^{W} e^{-F_\lambda^{ij}/T_\lambda}} \tag{12}$$

where F_λ^{ij} is the value of $K_{ij} \times \mathrm{EDV}(i)$, which means the fitness value of the jth weapon assigned to the ith target at iteration λ. Thereafter, if an elitist solution is found then it is updated and the pheromone trail update rule is applied accordingly. These steps are repeated until a stop criterion, such as enough number of iterations being performed or after a fixed running time, is satisfied.

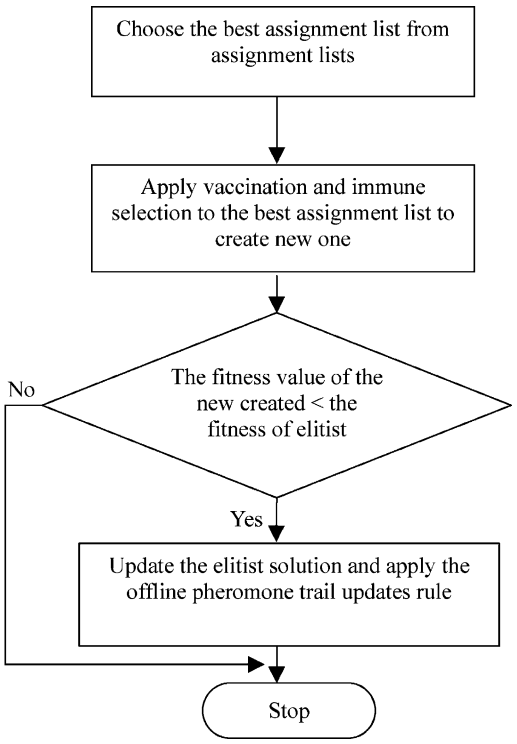
Choose the best assignment list from assignment lists
Apply vaccination and immune selection to the best assignment list to create new one
The fitness value of the new created < the fitness of elitist
No
Yes
Update the elitist solution and apply the offline pheromone trail updates rule
Stop

Fig. 2. The diagram of immune system.

6. Simulation and results

In our implementation, the following parameters are set as default values: $\rho = 0.1$, $\psi = 0.1$, $q_0 = 0.8$, $\beta = 2$, and m is the maximum number of weapons or targets. The first scenario with 10 targets and 10 weapons is considered to test the performance of various local search mechanisms used in ACO. The simulation results are listed in Table 1. Note that the results are averaged over 10 trials to smooth out random characteristics in simulations conducted. It is evident that these approaches can 100% converge to the best fitness value. Nevertheless, the overall performance for ACO with an immune system as local search can possess the best search efficiency.

Table 1
The simulation results for randomized data of $W = 10$ and $T = 10$

Algorithms	Best fitness value	Convergence (%)	CPU time (s)
ACO with general local search	58.4774	100	21.4808
ACO with SA as local search	58.4774	100	28.7341
The proposed algorithm	58.4774	100	20.1665

Results are averaged over 10 trials.

Table 2
Compare the best fitness values obtained by various search algorithms

Algorithms	$W = 50$, $T = 50$	$W = 80$, $T = 80$	$W = 100$, $T = 80$	$W = 120$, $T = 80$
SA	290.4569 (50.8541)	392.5413 (50.8137)	290.564 (43.8612)	175.631 (35.1352)
GA	282.65 (47.6715)	351.7641 (52.9217)	279.558 (50.4279)	140.1381 (38.2817)
GA with general local search	226.335 (30.7316)	348.4352 (60.7415)	197.8655 (37.1495)	106.4778 (12.9815)
GA with immunity as local search	171.8513 (25.8734)	282.2294 (30.2742)	161.3612 (19.0139)	98.8892 (8.1775)
ACS with general local search	193.0035 (10.2913)	277.2656 (15.8872)	169.6705 (9.0167)	70.2907 (6.2714)
ACS with SA as local search	148.5712 (6.8915)	204.2938 (7.9671)	103.8276 (5.8216)	63.8636 (5.3581)
The proposed algorithm	139.2332 (5.6347)	195.2245 (7.8912)	96.9271 (5.3867)	52.2157 (4.1579)

Results are averaged over 10 trials.

Now, other scenarios are considered. The used search algorithms are ACO, SA and genetic algorithm (GA). GA is a search algorithm based on the concept of natural selection. In our GA implementation, the assignments are represented as genes on a chromosome [25]. In this implementation, the traditional one-cut-point crossover, inversion mutation, and roulette selection operators are implemented for the GA [33,34]. These parameters used are the crossover probability $P_c = 0.8$, the mutation probability $P_m = 0.4$ in GA, and $\gamma = 0.5$ in SA. The simulation is also conducted for 10 trials. Since those algorithms are search algorithms, it is not easy to stop their search in a fair basis from the algorithm. In our comparison, we simply stopped those algorithms after a fixed time period of running. Experiments were conducted on PCs with PIII 1 GHz processors, and were stopped after 2 h of running. The results of averaged best fitness values and standard deviation in parentheses are listed in Table 2. From the comparison, it is again evident that the proposed algorithm can have best search efficiency for all examples among all algorithms.

Furthermore, we want to test the convergent performance. This scenario consists of a set of randomized data for 120 weapons and 100 targets. In our study, we tried to find when those algorithms converged to the optimal solution. The found best fitness values and the time periods for those algorithms to find their best fitness values are listed in Table 3. Also, the fitness curves by iterations (or generations) are showed in Fig. 3. Note that the computational times required for each generation are different for different algorithms. Evidently, in our simulation, those curves are similar and can all converge to the same fixed value. This is because they all use the same ACS framework. From Table 3, it is evident that proposed algorithm used the least time to converge to the optimal solution among those algorithms. Even though the time difference in the shown example is not significant, from the results in Table 2, the search efficiency of the proposed algorithm still is obvious.

Table 3
The simulation results for randomized data of $W = 120$ and $T = 100$

Algorithm	Best fitness	CPU time (min)
GA with general local search	173.3241	335.3
GA with immunity as local search	173.3241	285.4
ACS with general local search	173.3241	273.4
ACS with SA as local search	173.3241	139.3
The proposed algorithm	173.3241	136.3

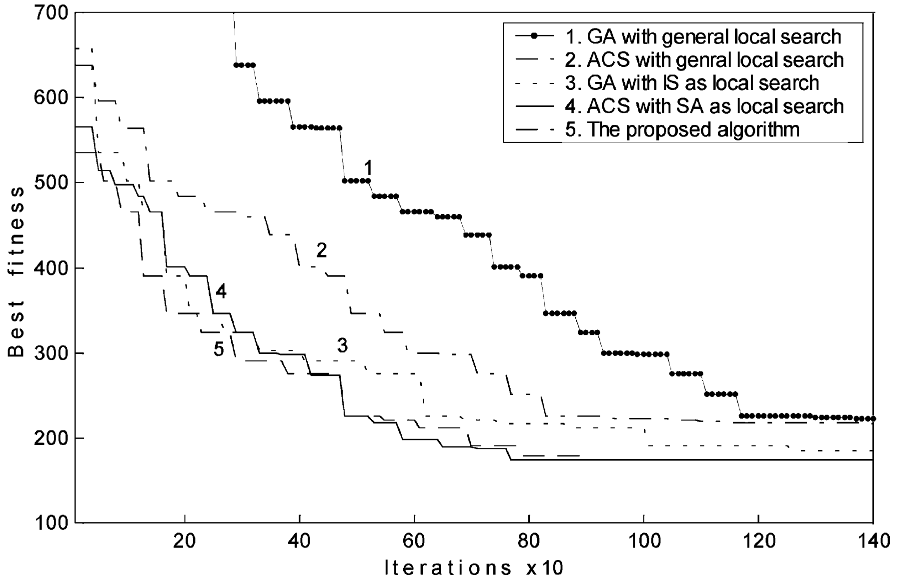

Fig. 3. The fitness curves of $W = 120$ and $T = 100$.

7. Conclusions

In this paper, we proposed an immunity-based ACO algorithm. It consists of two types of search algorithms, the ACO and the immune system (IS). From our simulations for those WTA problems, the proposed algorithm indeed has better search performances than other existing algorithms do.

Acknowledgements

The authors would like to thank the anonymous reviewers for their comments and constructive suggestions that have improved the paper. This work was supported in part by the Chung-Shan Institute of Science Technology of Taiwan under the grant XU89A59P and in part by the National Science Council of Taiwan, ROC under the grant NSC-89-2218-E-011-002.

References

[1] S.P. Lloyd, H.S. Witsenhausen, Weapon allocation is NP-complete, in: Proceedings of the IEEE Summer Simulation Conference, 1986.

[2] T. Bäck, U. Hammel, H.-P. Schwefel, Evolutionary computation: comments on the history and current state, Proc. IEEE Trans. Evolut. Computat. 1 (1) (1997) 3–17.

[3] V.J. Rayward-Smith, I.H. Osman, C.R. Reeves, G.D. Smith, Modern Heuristic Search Methods, Wiley, New York, 1996.

[4] R. Beckers, J.L. Deneubourg, S. Goss, Trails and U-turns in the selection of the shortest path by the ant *Lasius niger*, J. Theor. Biol. 159 (1992) 397–415.

[5] S. Goss, S. Aron, J.L. Deneubourg, J.M. Pasteels, Self-organized shortcuts in the argentine ant, Naturwissenchafte 76 (1989) 579–581.

[6] M. Dorigo, V. Maniezzo, A. Colorni, The ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybernet., Part B 26 (1996) 29–41.

[7] V. Maniezzo, A. Colorni, The ant system applied to the quadratic assignment problem, IEEE Trans. Knowledge Data Eng. 11 (1999) 769–778.

[8] T. Stutzle, H. Hoos, MAX–MIN ant system and local search for the traveling salesman problem, in: Proceedings of the IEEE International Conference on Evolutionary Computation, 1997, pp. 299–314.

[9] A.M.H. Bjorndal, et al., Some thoughts on combinatorial optimization, Eur. J. Operational Res. 13 (1995) 253–270.

[10] T. Ibarraki, N. Katoh, Resource Allocation Problems, MIT Press, Cambridge, MA, 1988.

[11] P.L. Hammer, Some network flow problems solved with pseudo-Boolean programming, Operational Res. 13 (1965) 388–399.

[12] D.L. Pepyne, et al., A decision aid for theater missile defense, in: Proceedings of 1997 IEEE International Conference on Evolutionary Computation, 1997.

[13] H.A. Eiselt, G. Laporate, A combinatorial optimization problem arising in dartboard design, J. Operational Res. Soc. 42 (1991) 113–181.
[14] S. Sahni, T. Gonzales, P-complete approximation problem, ACM J. 23 (1976) 556–565.
[15] A. Bauer, et al., An ant colony optimization approach for the single machine total tardiness problem, in: Proceedings of the 1999 Congress on Evolutionary Computation, Vol. 2, 1999, pp. 1445–1450.
[16] G.N. Varela, M.C. Sinclair, Ant colony optimization for virtual wavelength path routing and wavelength allocation, in: Proceedings of the 1999 Congress on Evolutionary Computation, Vol. 3, 1999, pp. 1809–1816.
[17] G. Di Caro, M. Dorigo, Mobile agents for adaptive routing, in: Proceedings of the 31st Hawaii International Conference on System Sciences, Vol. 7, 1998, pp. 74–83.
[18] I.K. Yu, C.S. Chou, Y.H. Song, Application of the ant colony search algorithm to short-term generation scheduling problem of thermal units, in: Proceedings of the 1998 International Conference on Power System Technology, Vol. 1, 1998, pp. 552–556.
[19] L. Nemes, T. Roska, A CNN model of oscillation and chaos in ant colonies: a case study, IEEE Trans. Circuits Syst. I: Fundam. Theory Applic. 42 (10) (1995) 741–745.
[20] F. Abbattista, N. Abbattista, L. Caponetti, An evolutionary and cooperative agents model for optimisation, in: Proceedings of the IEEE International Conference on Evolutionary Computation, Vol. 2, 1995, pp. 668–671.
[21] P.J.M. Van Laarhoven, E.H.L. Arts, Simulated Annealing: Theory and Applications, Kluwer Academic Publishers, Dordrecht, 1992.
[22] P.P.C. Yip, Y.-H. Pao, A guided evolutionary simulated annealing approach to the quadratic assignment problem, IEEE Trans. Syst. Man Cybernet. 24 (1994) 1383–1386.
[23] M. Dorigo, G. Di Caro, Ant colony optimization: a new meta-heuristic, in: Proceedings of the 1999 Congress on Evolutionary Computation, Vol. 2, 1999, pp. 1470-1477.
[24] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence from Natural to Artificial Systems, Oxford University Press, Oxford, 1999.
[25] Z.-J. Lee, C.-Y. Lee, S.-F. Su, A fuzzy-genetic based decision-aided system for the naval weapon–target assignment problems, in: Proceedings of the 2000 ROC Automatic Control Conference, 2000, pp. 163–168.
[26] A. Colorni, M. Dorigo, V. Maniezzo, Distributed optimization by ant colonies, in: Proceedings of European Conference on Artificial Life (ECAL'91), Elsevier, Amsterdam, 1991.
[27] A. Colorni, M. Dorigo, V. Maniezzo, An investigation of some properties of an ant algorithm, in: Proceedings of the Parallel Problem Solving from Nature Conference, Elsevier, Amsterdam, 1992.
[28] S.A. Frank, in: M.R. Rose, G.V. Lauder (Eds.), The Design of Natural and Artificial Adaptive Systems, Academic Press, New York, 1996.
[29] D. Dasgupta, N. Attoh-Okine, Immunity-based systems: a survey, in: Proceedings of the IEEE International Conference on Computat. Cybernet. Simul. 1 (1997) 369–374.
[30] I. Tazawa, S. Koakutsu, H. Hirata, An immunity-based genetic algorithm and its application to the VLSI floor plan design problem, in: Proceedings of IEEE International Conference on Evolutionary Computation, 1996, pp. 417–421.
[31] A. Gasper, P. Collard, From GAs to artificial immune systems: improving adaptation in time-dependent optimization, in: Proceedings of the 1999 Congress on Evolutionary Computation, Vol. 3, 1999, pp. 1999–1866.
[32] P. Merz, B. Freisleben, Fitness landscape analysis and memetic algorithms for quadratic assignment problem, IEEE Trans. Evolut. Computat. 4 (4) (2000) 337–352.
[33] M. Gen, R. Cheng, Genetic Algorithms and Engineering Design, Wiley, New York, 1997.
[34] D. Goldberg, R. Lingle, Alleles, loci and the traveling salesman problem, in: Proceeding of the first International Conference on Genetic Algorithm, Lawrence Erlbaum, Hillsdale, NJ, 1985, pp. 154–159.
[35] D.J. Rosenkrantz, R.E. Stearns, P.M. Lewis, An analysis of several heuristics for the traveling salesman problem, SIAM J. Comput. 6 (1997) 563–581.
[36] L. Jiao, L. Wang, Novel genetic algorithm based on immunity, IEEE Trans. Syst. Man Cybernet., Part A 30 (5) (2000) 552–561.
[37] E.H.L Aarts, J.K. Lenstra, Local Search in Combinatorial Optimisation, Wiley, New York, 1997.
[38] S. Kirkpatrick, C. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 220 (1983) 671–680.
[39] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. 21 (1953) 1087–1092.
[40] J.-S.R. Jang, C.-T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing, Prentice-Hall, Englewood Cliffs, NJ, 1997.
[41] M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evolut. Computat. 1 (1) (1997) 53–66.
[42] Z.-J. Lee, S.-F. Su, C.-Y. Lee, A genetic algorithm with domain knowledge for weapon–target assignment problems, J. Chinese Inst. Engrs. 25 (3) (2002) 287–295.