Chapter 7
Modified Ant Colony Optimization Algorithm for the Multi-Sensor Dynamic Scheduling

Hai Huang, Jing Zhang, Xiaomin Ran and Wengao Lv

Abstract Sensor is a sort of important monitoring resources and plays an irreplaceable role in the modern battlefield. Multi-sensor scheduling optimization is a problem of theoretical and practical significance. In order to monitor the multi-target with time windows effectively, this paper presents a multi-sensor dynamic scheduling model and demonstrates its reasonableness. Based on the model, we adopt a modified Ant Colony Optimization (ACO) algorithm with local optimization method to find optimal solutions, and conduct several experiments under different scenarios. The results show that more targets are monitored effectively in each solution, therefore the modified ACO algorithm has better performance than basic ACO algorithm in scheduling optimization.

Keywords Ant colony optimization (ACO) · Multi-sensor · Dynamic scheduling · Time window · Local optimization

7.1 Introduction

With the development of network and information technology, a variety of sensors is widely used on the battlefield, multi-sensor scheduling subsequently becomes an important problem. Multi-sensor scheduling try to fulfill a series of monitoring tasks using a given set of sensors. The number of sensor and target creates a huge number of combinations to be searched for producing a scheduling solution that optimizes appraisal indices such as monitoring quality and sensor utilization. In essence, it is a combinatorial optimization problem that has been proven to be a NP-hard problem. Exact optimization algorithms cannot produce an acceptable

H. Huang (✉) · J. Zhang · X. Ran · W. Lv
Zhengzhou Information Science and Technology Institute, Zhengzhou, Henan, China
e-mail: huanghai11111@sina.com

solution in the available time. It is also awkward to deal with some necessary constraints or objective function characteristics in the algebraic form required by classical optimization methods. Therefore, the algorithm cannot meet the requirements of practical application. Moreover, when the targets with feature of time window, static scheduling cannot make full use of sensor resources and a large number of missions cannot be completed.

In the field of research on sensor dynamic scheduling, Liu [1] proposed a sensor management method based on utility function. It obtains the optimal solution by establishing a linear programming objective function that relates to effective matching function and target priority. Although this method is simple and feasible, the amount of computing is too large to achieve large-scale scheduling. Xiao and Xiao [2, 3] proposed an incremental sensor selection heuristic algorithm and introduced the Monte Carlo operator to calculate the detection probability of target and schedule the sensors under the condition that the sensor must meet the effective detection probability demand. However, this method only refers to a single target tracking tasks. Considering the characteristics of multi-sensor multi-target assignment, Zhang et al. [4] presented a sensor scheduling method amid at Joint Information incremental performance. The method is combined with Genetic Algorithm. However, this method only focuses on the overall incremental information. Besides, all the methods mentioned above are not involved in the target problem of time window.

In this paper, we present a sensor dynamic scheduling model that adapted to the scheduling environments of multi-sensor and multi-target with time window. It takes global coverage of targets as the optimization objective. Then we introduce an approach to scheduling that relies on a biologically-inspired optimization algorithm known as Ant Colony Optimization (ACO). Moreover, the algorithm has been modified. The simulation results show that the improved ACO is more effective than basic ACO. The results are simply intended to be a proof that ACO can successfully deal with the challenge we describe.

7.2 Models and Problem Statement

$U = \{u_1, u_2 \ldots, u_M\}$ denotes a monitoring system with M sensors. The target set $T = \{t_1, t_2 \ldots, t_N\}$ is composed of N targets. A real-time task to monitor t_i is represented by the tuple (tb_i, te_i), where tb_i is the task start time and te_i is the task end time. v_{ij} denotes the monitoring ability when t_j is monitored by u_i. d_{ij} denotes the distance between u_i and t_j. $e_i$denotes the probability that t_i can be detected. It is related to the size, shape and velocity of target. Therefore, the monitoring ability v_{ij} is almost linearly related to the product of e_i and ${}^{1}\big/_{d_{ij}^2}$: $v_{ij} \propto e_j \times {}^{1}\big/_{d_{ij}^2}$. g_i denotes the threshold which is the lowest value when u_i can monitor target effectively. It is determined by the sensitivity of sensor.

In order to utilize resource fully and monitor as more targets as possible, we take global coverage of targets as the optimization objective, rather than the usual total monitoring quality. So this problem can be described as a mathematical programming model as shown below:

$$\max \quad f(X) = \sum_{i=1}^{M} \sum_{j=1}^{N} x_{ij} \Big/ N \tag{7.1}$$

s.t.

$$\sum_{i=1}^{M} x_{ij} \leq 1, \quad j = 1, 2, \ldots\ldots, N \tag{7.2}$$

$$x_{ij} = 1, u_i \in U, t_i \in T \Rightarrow v_{ij} \geq g_i \tag{7.3}$$

$$\forall i, j, t_i \in \mathrm{T}, \mathrm{t_j} \in \mathrm{T}, \mathrm{u}_k \in U, \text{if } x_{ki} = x_{kj} = 1 \Rightarrow tb_i \geq te_j \; or \; tb_j \geq te_i \tag{7.4}$$

Where $X = (x_{ij})_{M \times N}$ is the decision matrix and the x_{ij} is as follows:

$$x_{ij} = \begin{cases} 1, & if \ sensor\, i\, is\, to\, target\, j \\ 0, & otherwise \end{cases} \tag{7.5}$$

As stated in constraint given in Eq. 7.2, the target cannot be monitored by multiple sensors. It means that some targets can be abandoned for the global optimization. Eq. 7.3 explains the requirement of the effective monitoring: the monitoring quality v_{ij} must be greater than or equal to the ability threshold of u_i when t_j is assigned to u_i. Meanwhile, Eq. 7.4 means that one sensor can monitor only one target simultaneously, but can monitor other target at non-intersecting time window. The two constraints cannot only avoid invalid monitoring, but also increase resource utilization.

According to the analysis above, the dynamic scheduling problem is abstracted to a combination problem with optimization objectives and complex constraints. Based on the model, we will adopt the method combined with Ant Colony Optimization (ACO) algorithm and local optimization to solve the problem effectively. The details are as follows.

7.3 The Solving Method for Multi-Sensor Dynamic Scheduling

7.3.1 ACO Background

The Ant Colony Optimization was inspired by the behavior of ants. Ants communicate among themselves through pheromone, a substance they deposit on the ground in variable amounts as they move about. It has been observed that the more ants use a particular path, the more pheromone is deposited on that path and the more it becomes attractive to other ants seeking food. The pheromone on the shorter path will therefore be more strongly reinforced and will eventually become the preferred route for the other ants. The works of Colomi et al. [5] offer detailed information on the workings of the algorithm and the choice of the various parameters. Moreover, the ACO algorithm has been applied to many classical problem such as TSP [6], VRP [7], JSP [8] and so on.

7.3.2 Modified ACO Algorithm for the Problem

We use a digraph based on the ACO algorithm [9, 10] to treat the complex problem that we have described. As shown in Fig. 7.1, each node except starting node represents a target and each path in front of node represents a sensor. Each ant establishes a complete tour in a digraph (a feasible solution) by repeatedly choosing path to next node in probability. The probability is a combination of heuristic and pheromone information. To ensure the production of a feasible solution, paths that ant cannot go through because of constraints on the current tour are excluded from the choice by using a tabu list. After a tour, ant leaves the pheromone on the path. The pheromone trails reinforce successful paths discovered by the ants so that those paths are more likely to be followed in future tours. The algorithm will end and output the best result when it gets max iteration. We will explain the key steps as follow.

7.3.3 Trail Selection Probability

Ant $k = (1, 2, \ldots\ldots N_{ants})$ selects moving path according to probability in the process of solution construction. $p_{ij}^{k}(t)$ is the probability that an ant will assign task j to sensor i ant at time t:

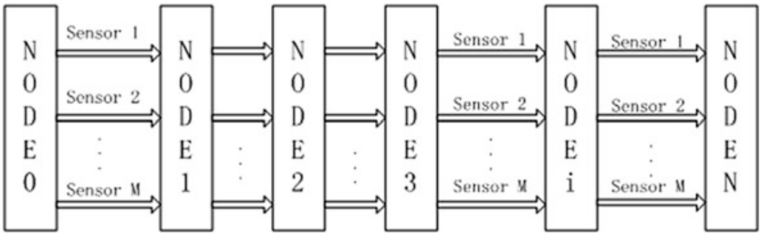

Fig. 7.1 Digraph based on the ACO algorithm

$$\mathrm{p}_{ij}^{k}(t) = \begin{cases} \dfrac{\tau_{ij}^{k}(t)^{\alpha}(\eta_{ij})^{\beta}}{\sum\limits_{i \notin tabu_j^k(t)} \tau_{ij}^{k}(t)^{\alpha}(\eta_{ij})^{\beta}}, & i \notin tabu_j^k(t) \\ 0, \quad i \in tabu_j^k(t) \end{cases} \tag{7.6}$$

For the selection of a path, the ant uses heuristic information as well as pheromone information. The heuristic information, denoted by η_{ij} and the pheromone information, denoted by $\tau_{ij}^{k}(t)$. Initial pheromone in each path isτ_0, where α, β denote the parameters correlating to the importance of the pheromone and heuristic, respectively. $tabu_j^k(t)$ is the tabu list of ant k that represents the sensors can not assigned to target j. η_{ij} is shown by Eq. 7.7:

$$\eta_{ij} = \begin{cases} {}^{v_{ij}}\!/_{g_i}, & v_{ij} \geq g_i \\ 0, & v_{ij} < g_i \end{cases} \tag{7.7}$$

7.3.4 Pheromone Update

7.3.4.1 Local Update Rule

The pheromones $\tau_{ij}^{k}(t)$ are updated by the local updating rule after an ant has one solution. The modified ACO adopts the following local updating rule to prevent succeeding ants from searching in the neighborhood of the current schedule of the current ant. The pheromone levels are modified as follows:

$$\tau_{\mathrm{ij}}^{k+1}(t) = (1-\rho)\tau_{ij}^{\mathrm{k}}(t) + \rho\Delta\tau, if(i,j) \in \pi_k \tag{7.8}$$

Where $\Delta\tau$ represents local pheromone increment and is constant. ρ is the local evaporation rate of pheromone trails. π_k is the solution of ant k.

7.3.4.2 Global Update Rule

After all ants have built all feasible schedules, the global update rule, Eq. (7.10) is used to increase the pheromone $\tau_{ij}^{k}(t)$ by applying the best solution in this iteration. For all $\tau_{ij}^{k}(t)$, the pheromone is increased by the global update and evaporated by global pheromone evaporation rate, as shown in Eq. (7.10).

$$\tau_{ij}(t+1) = (1-\lambda)\tau_{ij}^{N_{ants}}(t) + \Delta\tau_{ij}^{*}(t), \lambda \in (0,1) \tag{7.9}$$

$$\Delta\tau_{ij}^{*}(t) = \begin{cases} Q \times BestValue, & if(i,j) \in \pi^{*}(t) \\ 0, & else \end{cases} \tag{7.10}$$

Where λ is the global evaporation rate of pheromone trails. $\Delta\tau_{ij}^{*}(t)$ is the pheromone of best solution in the t th iteration. Q represents local pheromone increment and is constant. N_{ants} is the max number. *BestValue* is the best solution in *t*th iteration when the algorithm runs.

This is an elitiststrategy that leads ants to search near the best-foundsolution. Using the best ant for updating makes the search much more aggressive. Best combinations which often occur in good solutions will get a lot of reinforcement. Therefore, the algorithm has some extra features to balance exploration versus exploitation. So we must make choice between using the iteration-best ant and the global-best. Using global-best results in strong exploitation will lead to quick convergency of algorithm, so we will alternate it with the use of iteration-best.

7.3.5 Local Optimization Strategy

It is known that the performance of ACO algorithms can sometimes be greatly improved when coupled to local search algorithms [11]. What normally happens is that a population of solutions is created using ACO, and then these solutions are improved via local search. The improved solutions are then used to update the pheromone trail, so it is in fact a form of Lamarckian search.

In the our version of the ACO algorithm, the best solution is improved through local optimisation phase.In this phase, one of the targets that have been monitored frees its sensor. Then, for every remaining targets that no sensor monitors it, it is investigated whether the sensors can be allocated to it. If it is possible, the algotithm assigns one of the feasible sensors to this targets randomly. The algorithm successively tries to make more other vacant targets get sonor. So, a complete new solution is created. If the result of new solution is better than the best solution before, the complete local search procedure is then repeated with this new solution and updates the pheromone trail globally. This procedure is iterated until no improvement in fitness is detected between the solutions before and after the local search is applied. Hence, the local search is in fact a hill-climbing algorithm which takes the original solution created by the ACO procedure to the nearest local optimum.

7.4 Experimental Simulations

To verify the validity of our proposed methods, we conduct two experiments. After many simulations, we get optimal parameters and set it in the experiments: $\rho = 0.9$, $\alpha = 2$, $\beta = 1$, $\tau_0 = 1$, $\lambda = 0.85$, Q $= 10$, $\Delta\tau = 1.5$, $N_{ants} = 20$, $NC_{\max} = 100$.

Table 7.1 The targets' time information of experiment 1

Target	1	2	3	4	5	6	7	8	9	10	11	12
StartTime	41	184	182	135	89	115	122	183	154	173	52	114
EndTime	55	196	198	146	100	127	132	195	169	188	67	124

Experiment 1: We test the improved ACO algorithm with data of small scale. Suppose there are 5 sensors and 12 targets in this numerical example. Table 7.1 shows the time information of the targets. Table 7.2 shows the capacity that every sensor monitors the targets. Table 7.3 shows the scheduling scheme result of modified ACO. The result shows that the algorithm is effective to solve this problem. The scheduling scheme is reasonable and makes full use of multi-sensor. Almost all targets can be monitored effectively and timely.

Experiment 2: The modified ACO algorithm is tested on the different size problem and compared with basic ACO algorithm. We generate four groups of data randomly to test the algorithm we introduce. The capacity v_{ij} is a integer value set to a uniform random number in $[0, 25]$. g_i is a integer value set to a uniform random number in $[15, 20]$. Scheduling time is between 0 and 200. The period of task $te_i - tb_i$ is set to a uniform random number in $[10, 30]$.

Table 7.4 shows the simulation results of modified ACO and basic ACO. Each case was simulated 100 times; each simulation as set to run for 100 iterations. Simulation results indicate that the modified ACO can find optimal solution for the problems of different size. Moreover, when the problem size is small, modified algorithm runs faster to find optimal solution. In the problem of large size, modified ACO can reach better value than basic ACO though needs a little more iterations. Actually, we can accept time consumption the modified ACO makes.

Figure 7.2 shows the global-best value of modified ACO and basic ACO for cases of 35 sensors and 150 targets in best simulation result. It illustrates that basic ACO is trapped in local optimum.However, modified ACO avoid immature convergence and the global search ability of the algorithm is strengthened to escape from local optimum and approach the global optimum.

Table 7.2 The monitoring quality matrix of experiment 1

Target sensor	1	2	3	4	5	6	7	8	9	10	11	12
1	7	5	2	11	8	0	2	4	6	10	3	24
2	17	12	5	13	15	23	20	17	20	21	10	22
3	10	7	8	19	21	6	7	3	3	3	24	18
4	20	13	24	25	24	14	11	13	17	17	19	15
5	8	25	5	18	6	14	18	14	17	20	21	23

Table 7.3 Scheduling result of experiment 1

Target	1	2	3	4	5	6	7	8	9	10	11	12
Sensor	4	5	4	4	3	4	2	–	2	2	4	1

Table 7.4 Comparison of modified ACO with basic ACO in different problem size

Problem size	Modified ACO			Basic ACO		
	Best-value (%)	Best-iteration	Avg-value (%)	Best-value (%)	Best-Iteration	Avg-value (%)
M = 15, N = 50	94	17	90.6	94	19	90.3
M = 15, N = 100	78	26	72.4	72	29	69.2
M = 20, N = 100	84	39	81.1	77	36	75.8
M = 35, N = 150	92.67	50	87.3	90	47	81.7

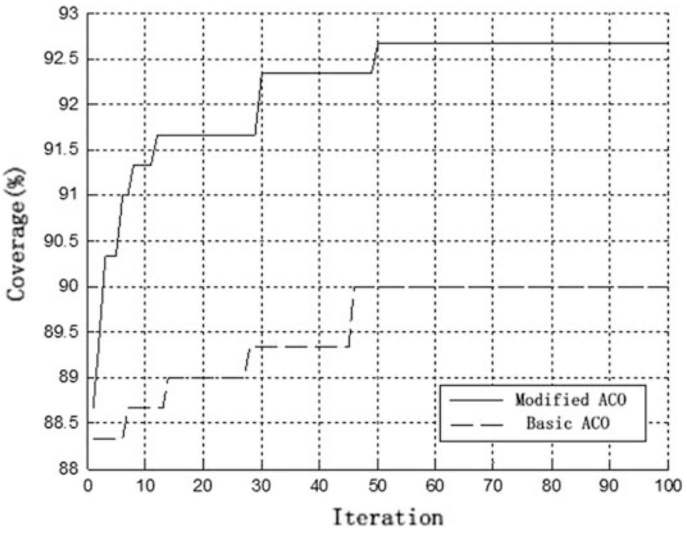

Fig. 7.2 Global-best value for cases of 35 sensors and 150 targets

7.5 Conclusion and Future Work

In this paper, we propose a sensor dynamic scheduling model considering time windows of targets. The global coverage of targets and several constraints are introduced to the model. We also present a hybrid ACO algorithm combined with local optimization method to solve the problem. Preliminary test shows that this approach is appropriate for multi-sensor dynamic scheduling and has better performance than basic ACO algorithm. Experimental results regarding the reduction of energy consumption will be presented in a future paper. We also plan to investigate the impact of heuristic on the performance of our ACO algorithm.

References

1. Liu X (2000) Study on algorithm of sensor management based on functions of efficiency and waste. Chin J Aeronaut 13(1):39–44 (in Chinese)
2. Xiao W, Wu J, Xie L et al (2006) Sensor scheduling for target tracking in networks of active sensors. ACTA Automatica Sinica 32(6):922–928
3. Xiao W et al (2006) Multi-sensor scheduling for reliable target tracking in wireless sensor networks. In: International conference on its telecommunications proceedings, pp 996–1000
4. Zhang G, Wang F, Wei Z (2008) Sensor management algorithm based on genetic algorithm. Mod Defence Technol 36(6):91–95
5. Colomi A, Dorigo M, Maniezzo V (1991) Distributed optimization by ant colonies. Proceedings of the first European conference on artificial life, Paris, France, pp 134–142
6. You X et al (2009) On multi-behavior based multi-colony ant algorithm for TSP. Intell Inf Technol Appli, pp 178–189
7. Huai-long, Hua D (2010) Vehicle routing problem of logistics based on dynamic ant colony algorithm. Educ Technol Comput Sci (ETCS) 256–262
8. Cao Y, Song X (2009) A hybrid algorithm of converse ant colony optimization for solving JSP. Comput Intel Soft Eng 234–240
9. Haibin D (2005) The theory and application of ant colony algorithm. Science Press, Bejing, pp 745–752 (in Chinese)
10. Zong-yong L, Xia P, Zhixue W, Ying L (2007) Scheduling interrelated tasks in grid based on ant algorithm. J Syst Simul 6:3196–3199 (in Chinese)
11. Dorigo M, Stützle T (2001) The ant colony optimization metaheuristic: algorithms, applications, and advances. Handbook of metaheuristics. In: Glover F, Kochenberger G (eds) pp 733–742