Science in China Series F: Information Sciences

Springer

www.scichina.com
info.scichina.com
www.springer.com/scp
www.springerlink.com

Evolutionary decision-makings for the dynamic weapon-target assignment problem

CHEN Jie[1,2], XIN Bin[1,2†], PENG ZhiHong[1,2], DOU LiHua[1,2] & ZHANG Juan[1,2]

[1] School of Automation, Beijing Institute of Technology, Beijing 100081, China;

[2] Key Laboratory of Complex System Intelligent Control and Decision, Ministry of Education, Beijing 100081, China

The dynamic weapon-target assignment (DWTA) problem is an important issue in the field of military command and control. An asset-based DWTA optimization model was proposed with four kinds of constraints considered, including capability constraints, strategy constraints, resource constraints and engagement feasibility constraints. A general "virtual" representation of decisions was presented to facilitate the generation of feasible decisions. The representation is in essence the permutation of all assignment pairs. A construction procedure converts the permutations into real feasible decisions. In order to solve this problem, three evolutionary decision-making algorithms, including a genetic algorithm and two memetic algorithms, were developed. Experimental results show that the memetic algorithm based on greedy local search can generate obviously better DWTA decisions, especially for large-scale problems, than the genetic algorithm and the memetic algorithm based on steepest local search.

decision-making, dynamic weapon-target assignment (DWTA), military command and control, evolutionary computation, memetic algorithms, constraints handling

1 Introduction

The dynamic weapon-target assignment (DWTA) problem is a typical constrained combinatorial optimization problem arising in the field of military operations research. Its objective is to minimize the expected damage of own-force assets by assigning available weapons to offensive targets at appropriate occasions. In fact, the weapon-target assignment (WTA) problem is one of the crucial issues for the automation of military command and control (C2) which is regarded as a challenge to control science[1]. WTA has been proved to be NP-Complete[2]. It has two versions — static WTA (SWTA) and dynamic WTA (DWTA)[3,4]. In SWTA, all weapons engage targets in a single stage, and all of the parameters for the problem are known. Thus, the goal of SWTA is to find the optimal assignment for a temporary defense task. In contrast, DWTA is a multi-stage problem and the outcome of each engagement is assessed for subsequent decisions. The goal of DWTA is to find a global optimal assignment for the whole defense process in which the engagement occasion of weapons must be taken into account. Intuitively, DWTA can be achieved by a series of SWTAs

Received January 29, 2009; accepted June 6, 2009
doi: 10.1007/s11432-009-0190-x

†Corresponding author (email: brucebin@bit.edu.cn)

Supported by the National Natural Science Foundation of China (Grant No. 60374069), and the Foundation of the Key Laboratory of Complex Systems and Intelligent Science, Institute of Automation, Chinese Academy of Sciences (Grant No. 20060104)

Citation: Chen J, Xin B, Peng Z H, et al. Evolutionary decision-makings for the dynamic weapon-target assignment problem. Sci China Ser F-Inf Sci, 2009, 52(11): 2006–2018, doi: 10.1007/s11432-009-0190-x

through all stages. However, although SWTA can at best guarantee the optimality of the WTA decisions for its corresponding defense stage, the combination of all SWTA decisions may not be optimal for the whole defense process. Besides, the actual issue of time windows which limit the engagement of weapons is not involved in SWTA[3]. In addition to the engagement constraints, the complexity of DWTA problems is also caused by resource constraints and strategy constraints[5,6].

Previous researches on WTA mainly focus on SWTA[6−20]. In respect of SWTA models, Hosein and Athans[3] proposed an asset-based SWTA model which was also used in refs. [7, 8]. Karasakal[9] used the probability of shooting down all incoming targets as the objective function of the air defense WTA model for a naval task group. Some scholars adopted target-based SWTA models which do not employ the value of assets directly[10−19]. Instead, each target in this case is assumed to have certain value of threat, and the objective is to minimize the total threat of all targets. In fact, the probability model in ref. [9] and the target-based models are just special cases of the asset-based model in ref. [3]. Besides, the cost of weapons is also taken into account in some models like that in ref. [20]. A more complicated model which considers the function of special assets can be found in ref. [21].

Based on the above models, varied algorithms have been proposed to solve SWTA since the middle of the last century. In the early stages, the algorithms for SWTA were limited to traditional algorithms such as implicit enumeration algorithms, branch-and-bound algorithms and dynamic programming[6]. With the development of computer technology, some novel algorithms such as neural networks[10], genetic algorithms (GAs)[7,11,12], taboo search (TS)[13], simulated annealing algorithms (SA)[14], ant colony optimization (ACO)[15], and particle swarm optimization (PSO)[16], have been developed. Some scholars also tried hybrid algorithms[8,17,18]. For example, Lee et al.[17] designed a memetic algorithm which combines the advantages of global search (genetic algorithm) and local search (greedy eugenics) to solve target-based SWTA problems. Besides, Ahuja et al.[19] developed several branch-and-bound algorithms and a very large scale neighborhood search algorithm to solve target-based SWTA problems.

Although nearly two decades have passed since DWTA was put forward[4], there are in contrast to SWTA only a few researches on DWTA[5,6,21−24]. Cai et al.[6] introduced some basic concepts on DWTA and provided a systematic survey on WTA problem. Hosein and Athans[4] did an early research on a two-stage asset-based DWTA problem and proposed a suboptimal algorithm for finding a good solution. Hosein et al.[21,22] also made some typical empirical experiments and provided analytical solutions to several simple cases of DWTA. Khosla[23] used a hybrid genetic algorithm which incorporates a simulated annealing-type heuristic to solve a target-based DWTA problem. In particular, a weighted combination of threat value and option weight is employed in the objective function of this DWTA model. Havens[24] models DWTA by means of simulation; however, it is in fact the repetition of SWTA[5]. Li et al.[5] proposed a target-based DWTA model with the objective of minimizing the total threat of the targets which survive the final stage of air defense operation. As far as we know, no algorithm has been proposed in the literature to solve asset-based DWTA problems with complicated constraints caused by the issue of engagement feasibility (e.g., the limit of time windows). The goal of this paper is to develop evolutionary decision-making algorithms to solve asset-based DWTA problems incorporating engagement feasibility.

The remainder is organized as follows. In section 2, the mathematical model for asset-based DWTA problem is formulated. A general "virtual" representation of solutions (decisions) is proposed to facilitate the generation of feasible solutions, and a constructive procedure transforms the virtual representation into a feasible solution. In section 3, a genetic algorithm and two memetic algorithms (MAs) for DWTA that are hybrid of genetic algorithms (GAs) and local improvement techniques are proposed. The results of employing different algorithms to solve DWTA problems are presented

and analyzed in section 4. Finally, section 5 concludes the paper.

2 Problem formulation

DWTA models depend on many factors such as defense strategies, features of targets and weapons and actual combat situations. Different defense scenarios may require different models. The scenario considered in this paper is narrated as follows. At certain time, the defender detects T offensive targets with their attack aims exposed, and K assets of the defender are threatened. There are W weapons available to intercept the targets. Before these targets break through the defense, there are at most S stages in which the defender's weapons can be assigned to certain targets. The value of S depends on the distance between targets and their aims, target's flight speed, weapon's regulation & launch & flight time, the delay of data analysis and decision-making, and so on[9]. A general engagement policy "Shoot-Look-Shoot (SLS)" is assumed which is a tradeoff between defense effect and defense cost[4,9,21−23].

2.1 Objective function

The total expected value of surviving assets after the final stage is taken as the objective of WTA for the current stage. This objective is somewhat similar to that in ref. [5] which also considers the optimization of the whole defense effect. The formulation of the objective function for stage t is shown as follows:

$$J_t(\boldsymbol{X}^t) = \sum_{k=1}^{K(t)} v_k \prod_{j=1}^{T(t)} \left[1 - q_{jk} \prod_{h=t}^{S} \prod_{i=1}^{W(t)} (1 - p_{ij}(h))^{x_{ij}(h)}\right] \quad \text{with} \quad t \in \{1, 2, \ldots, S\}, \tag{1}$$

where $W(t)$, $T(t)$ and $K(t)$ are, respectively, the total numbers of the remaining weapons, targets and assets in stage t; h is, like t, also an index of stage; $\boldsymbol{X}^t$=$[X_t, X_{t+1}, \ldots, X_S]$ with $X_t = [x_{ij}(t)]_{W\times T}$ is the decision variable in stage t, and $x_{ij}(t) = 1$ if weapon i is assigned to target j in stage t, $x_{ij}(t) = 0$ otherwise; v_k is the value of asset k; q_{jk} is the probability that target j destroys asset k; $p_{ij}(t)$ is the probability that weapon i destroys target j in stage t.

2.2 Constraints

The following four categories of constraints are included in our DWTA model.

$$\sum_{j=1}^{T} x_{ij}(t) \leqslant n_i, \forall t \in \{1, 2, \ldots, S\}, \quad \forall i \in \{1, 2, \ldots, W\}, \tag{2}$$

$$\sum_{i=1}^{W} x_{ij}(t) \leqslant m_j, \forall t \in \{1, 2, \ldots, S\}, \quad \forall j \in \{1, 2, \ldots, T\}, \tag{3}$$

$$\sum_{t=1}^{S} \sum_{j=1}^{T} x_{ij}(t) \leqslant N_i, \forall i \in \{1, 2, \ldots, W\}, \tag{4}$$

$$x_{ij}(t) \leqslant f_{ij}(t), \ \forall t \in \{1, 2, \ldots, S\}, \ \forall i \in \{1, 2, \ldots, W\}, \ \forall j \in \{1, 2, \ldots, T\}. \tag{5}$$

The constraint set (2) reflects the capability of weapons in firing at multiple targets at the same time. Most of actual weapons can shoot only one target at a time. Besides, a special weapon that can engage multiple targets simultaneously can be viewed as multiple separate weapons. In view of these facts, we set $n_i = 1$ for $\forall i \in \{1, 2, \ldots, W\}$. The constraint set (3) limits the weapon cost for each target in each stage. The setting of $m_j (j = 1, 2, \ldots, T)$ usually depends on the combat performance of available weapons. In our research, we suppose that $m_j = 1$ for $\forall j \in \{1, 2, \ldots, T\}$. This is a reasonable setting for missile-based defense systems and the "Shoot-Look-Shoot" engagement policy[5,9]. For artillery-based defense systems, the value of $m_j (j = 1, 2, \ldots, T)$ may be greatly increased under the same demand on defense strength. So the constraints in (3) can be considered as strategy constraints. The constraint set (4) reflects in essence the amount of ammunition equipped for weapons. $N_i (i = 1, 2, \ldots, W)$ is the maximal number of times that weapon i can be used due to the limit of its equipped ammunition. In the constraint set (5), $f_{ij}(t)$ is the indication of actual engagement feasibility for weapon i assigned to target j in stage t. $f_{ij}(t) = 0$ if weapon i cannot shoot target j in stage t with any potential reason; $f_{ij}(t) = 1$ otherwise. The time windows of targets

and weapons are the primary factors that affect engagement feasibility[5,6]. Constraint set (5) is very important to actual dynamic WTA problems since it contains the influence of time windows on the engagement feasibility of weapons. Besides, it also increases the complexity of DWTA problems and the difficulty of generating feasible solutions. In this case, it is hard to design a desirable operator which can generate new solutions and guarantee their feasibility at the same time.

2.3 Optimization model for DWTA

The optimization model for the dynamic WTA problem mentioned above is shown as follows:

$$\begin{aligned} &\text{Maximize} \quad J_t(\boldsymbol{X}^t) \\ &\text{s.t. } (2),(3),(4) \text{ and } (5), \\ &\text{with } t \in \{1,2,\ldots,S\}. \end{aligned} \tag{6}$$

It is clear that the DWTA problem is a constrained combinatorial optimization problem. In the research field of mathematical programming, it can also be categorized as a 0-1 programming problem. The dynamic characteristic of this model is mainly reflected by the stochastic nature of damage in each stage, the choice of engagement stages and the change of engagement feasibility. In contrast to target-based DWTA models[5,23], this asset-based model stresses on the protection of own-force assets especially those important assets, the ultimate goal and essence of WTA, as shown by its objective. Besides, the threats of all targets are directly contained in this model while target-based models depend on the evaluation of the threats of targets.

Note that we only consider the DWTA decision-making scenario corresponding to $t = 1$ for the comparison of different algorithms in this paper. In fact, decision-making will become relatively easier as t increases due to the destruction of targets and the consumption of weapons. There are no substantial differences between the DWTA decision-making scenarios corresponding to different stages.

2.4 Virtual representation and generation of feasible decisions

The variable $\boldsymbol{X}^t$ is a direct representation of DWTA decisions. However, it contains many immutable components restricted by the constraint set (5). In order to avoid violating the constraints in (5), we propose a virtual representation of DWTA decisions. The virtual representation is the permutation of all available assignment pairs. The definition of available assignment pairs is shown as follows.

Definition 1. Available assignment pair (AAP) If weapon i can be used to engage target j in stage t, then the assignment pair $i-j-t$ is an AAP. The pair $i-j-t$ is an APP iff $f_{ij}(t)=1$. Denote by S_{AAP} the set of all AAPs. It is easy to see that the engagement feasibility matrices $\boldsymbol{F}^t = [f_{ij}(t)]_{W\times T}(t=1,2,\ldots,S)$ determine S_{AAP}. So, $\boldsymbol{F}^t(t=1,2,\ldots,S)$ are utilized to get S_{AAP}. The following is an illustration for virtual representation.

Given

$$\boldsymbol{F}^1 = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad \boldsymbol{F}^2 = \begin{bmatrix} 0 & 1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

and

$$\boldsymbol{F}^3 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \end{bmatrix},$$

all AAPs with their indexes are shown as follows: (1) $i1-j1-t1$; (2) $i2-j1-t1$; (3) $i3-j2-t1$; (4) $i1-j2-t2$; (5) $i2-j2-t2$; (6) $i3-j1-t2$; (7) $i2-j1-t3$.

3-2-7-4-1-5-6, for instance, is a permutation of the seven AAPs that is a virtual representation of certain DWTA decision. In fact, the permutation-based representation is common for the quadratic assignment problem (QAP) which is a classical NP-hard combinatorial optimization problem[25]. However, the permutation employed for QAP is a direct representation of solutions which indicates the correspondence between facilities and locations. In contrast, the permutation for DWTA does not produce feasible decisions directly since it cannot guarantee the satisfaction of the constraints in (2), (3) and (4). In fact, the foundation on which we employ the virtual representation is that the generation of DWTA decisions depends on the assignment order of all AAPs. Besides, it is convenient to produce feasible decisions with such a representation

followed by a subsequent construction procedure. In this procedure, we present the concept of constraint saturation to deal with the constraints in (2), (3) and (4). Note that a constraint $c(\boldsymbol{X}) \leqslant 0$ is said to be saturated iff a decision $\boldsymbol{X}$ results in $c(\boldsymbol{X}) = 0$.

Initially, the states of all constraints in (2), (3) and (4) are unsaturated. In the constructive process, the states will be updated dynamically. The pseudocode for the construction procedure is shown in Table 1.

Since only AAPs are considered for the construction procedure, the constraints in (5) will not be violated. Thus, the generation of feasible decisions in the above procedure concentrates on the constraint satisfaction of (2), (3) and (4). It is evident that no violation of constraints will occur due to the mechanism of constraint saturation. Besides, each permutation can be used to produce only one feasible decision because the construction procedure is deterministic. All feasible decisions can be generated by certain permutations, but a feasible decision may not correspond to a unique permutation. In the following text on the design of evolutionary operators, we will elaborate some approaches to avoid generating the same decision repeatedly.

It should be noted that the increase of the use of any effective weapon without the violation of constraints will further improve the objective value and thus lead to a better decision. Therefore, the decision scheme in which all APPs are assigned will be optimal if it does not cause any violation of constraints. In general, however, the assignment of all APPs is infeasible due to the limitation of the constraints in (2), (3) and (4). Although the above construction procedure cannot ensure the optimality of generated decisions, it reduces the search scope of the DWTA problem greatly. This is because the generated decision with constraint saturations is the best feasible one among all feasible decisions generated by the same permutation with or without constraint saturations, and those inferior feasible decisions will not be produced.

3 Evolutionary decision-making algorithms

In the field of combinatorial optimization, it has been shown that combining evolutionary algorithms, e.g., genetic algorithms (GAs), with problem-specific heuristics can lead to more efficient approaches[17,26]. These hybrid algorithms

Table 1 Pseudocode for the construction procedure

Procedure construction (P_e) //P_e is a permutation of all AAPs (available assignment pairs; see Definition 1).

Initialize the saturation states of the constraints in (2), (3) and (4).

Let $\boldsymbol{X} = \boldsymbol{O}_{W\times T\times S}$ // a zero matrix

For $k = 1$ **to** size(S_{AAP})

If the assignment of the kth pair in the list L denoted by $i_k - j_k - t_k$ does not cause any violation of constraints

// L is the list formed by all AAPs in the order corresponding to the permutation P_e.

Let $x(i_k, j_k, t_k) = 1$

Update the saturation states of constraints related to the assigned AAP.

If all constraints in (2), (3) or (4) are saturated

Break //The procedure is over since any additional assignment will cause the violation of certain constraint.

End If

End If

End For

End Procedure

are often termed memetic algorithms (MAs), hybrid genetic algorithms, Lamarckian GAs, etc.[27].

In the framework of MAs, evolutionary algorithms are expected to perform the global search since they are usually competent for the global exploration of problem space. In contrast, problem-specific heuristics are responsible for local improvement. Thus, the synergy of global search and local refinement, as a primary effect of hybridization, contributes to the superiority of MAs. It has been approved that MA is one of the most popular and effective algorithms to solve combinatorial optimization problems like QAP[25,26]. So, it is rational to adopt MAs to solve the DWTA problem since it is also, as narrated above, a typical combinatorial optimization problem. In addition, we also employ genetic algorithms for performance comparisons. Typically, the common procedure of MAs developed on the framework of GAs can be described as follows:

Step 1. Initialize population (denote by Pop the initial population).

Step 2. Apply certain local refinement method to improve the initial population:

$$Pop = Local\_Refinement(Pop).$$

Step 3. Apply crossover operations on the population and use certain local refinement method to improve the solutions generated by crossover. Add the improved solutions into the population.

$$Pop = Pop \bigcup Local\_Refinement(Crossover(Pop)).$$

Step 4. Apply mutation operations on the population and use certain local refinement method to improve the solutions generated by mutation. Add the improved solutions into the population.

$$Pop = Pop \bigcup Local\_Refinement(Mutate(Pop)).$$

Step 5. Apply selection operations on the population to eliminate some bad solutions:

$$Pop = Select(Pop).$$

Step 6. If termination criteria are satisfied, stop the algorithm; otherwise, go to step 3.

The utilization of local refinement operations following initialization and all evolutionary operations except for selection, i.e., crossover and mutation, is the primary difference between MAs and their GA counterparts. So, the development of MAs mainly includes the design of local refinement approaches and that of GAs. In the following, we will present the content of the GA for DWTA and two local refinement approaches. Then, the resultant MAs can be constructed easily according to the above template for MA procedures. So, the details for MAs, for concision, will not be presented later. We will differentiate the MAs proposed in this paper by denominations later.

3.1 Local refinement approaches for DWTA

The goal of local refinement is to improve the current solution locally and even find a local optimum. Heuristic algorithms are often used to achieve local refinement, and there are mainly three kinds of heuristics — constructive methods, enumeration or limited enumeration, and improvement methods[25]. Constructive methods produce new solutions directly according to some predefined rules which usually contain some problem-specific knowledge. Enumeration can guarantee the optimality of generated solutions within certain neighborhood. However, it will generally take a long time to guarantee optimality when the search space is very large. In this case, a limited enumeration will be employed instead. The limited enumeration can reduce the execution time greatly; howbeit, it loses the guarantee of optimality. Improvement methods correspond to local search algorithms which are also termed as neighborhood search. An improvement method begins with a feasible solution and tries to improve it, searching for other solutions in its neighborhood. In the context of MAs, improvement methods are frequently applied. A very effective local search operator for many combinatorial optimization problems like QAP is the 2-opt operator which exchanges two selected elements in a permutation[26]. In our research, two local search methods based on a variant of the 2-opt heuristic are employed respectively for local refinement.

As a first step at introducing the proposed local refinement approaches, an illustrative example is shown in Figure 1 to describe the rationale of the basic operations in these approaches. As shown in Figure 1, weapon i_1 is assigned, as a result of the assignment of AAP5, to target j_1 in stage t_2. Thus, AAP8, if assigned, will make weapon i_1 assigned to two targets in the same stage and cause the violation of a constraint in (2). So it was discarded in the process of construction procedure. All assigned AAPs, abbr. AAAPs, form a feasible DWTA decision.

Any 2-opt operation within the set of AAAPs will not produce a new decision since any two AAAPs have no conflicts against each other in respect of constraint satisfaction. In fact, only when unassigned AAPs, abbr. UAAPs, are exchanged with AAAPs, is a new decision likely to be generated. However, the AAAPs for exchange have to be carefully chosen. For the purpose of ensuring the birth of new decisions, a UAAP should be exchanged with the AAAPs which collide with the UAAP regarding some constraints. These AAAPs can be regarded as the competitors of the UAAP to become the component of a feasible decision. All AAPs are categorized into two sets — S_{AAAP} and S_{UAAP} which, respectively, denote the set of AAAPs and that of UAAPs. Each 2-opt operation will select a UAAP from S_{UAAP} at first. Then, a competitive AAAP of the UAAP will be chosen for exchange. Further, it is required that the value of the AAAP is lower than that of the UAAP, which is another means to curtail computation time. Note that the value of an AAP denoted by $i-j-t$ is measured by the product of 1) the value of the asset threatened by target j, 2) the probability that target j destroys the asset, and 3) the probability that weapon i destroys target j. For example, the AAP8 in Figure 1 can exchange with AAP5 to produce a new decision. The neighborhood of the selective 2-opt operation is defined as the set of all permutations that can be formed by the current permutation using this operation. The operation is implemented iteratively according to two different local search strategies as described below.

1) Steepest strategy[27]: Find the best decision in the neighborhood of the current decision, and replace the current decision with it. A new local search proceeds in the neighborhood of the new decision.

2) Greedy strategy[27]: Once a better decision is detected in the neighborhood of the current decision, replace the current decision with it. A new local search proceeds in the neighborhood of the new decision.

The procedures of the two local search methods based on the above operator are shown in Table 2 and Table 3, respectively.

In contrast to the greedy local search procedure, the input variables of the steepest local search procedure, i.e., P_e and $\boldsymbol{X}$, are not changed during its implementation. It is easy to see that all solutions generated by the above two local search approaches are local optima. However, in our research, the neighborhood of a permutation will not be completely searched since it is usually very large. In order to reduce the complexity of local search approaches and make a better tradeoff between global

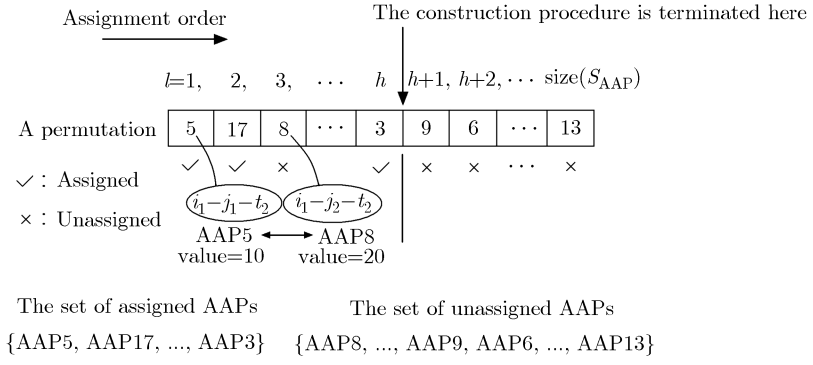

Figure 1 An illustration for the rational of basic operations.

Table 2 Pseudocode for the greedy local search procedure

Procedure Greedy_local_search(P_e, X)

//The input parameter P_e is a permutation of all AAPs, and $\boldsymbol{X}$ is the decision w.r.t. P_e.

//Denote by $\mathcal{N}(P_e)$ the neighborhood of P_e.

Repeat

Implement a selective 2-opt operation on P_e to get a new permutation $P_e^{'}$.

Apply the construction procedure to get the new decision $\boldsymbol{X}^{'}$ w.r.t. $P_e^{'}$.

If $\boldsymbol{J}(\boldsymbol{X}^{'}) > \boldsymbol{J}(\boldsymbol{X})$ **Then** $\boldsymbol{X} = \boldsymbol{X}^{'}$, $\boldsymbol{P_e}$=$\boldsymbol{P_e'}$ **End If**

Until $\forall\ \boldsymbol{P_e'} \in \boldsymbol{\mathcal{N}}\ (\boldsymbol{P_e})$: $\boldsymbol{J}(\boldsymbol{X}^{'}) \leqslant \boldsymbol{J}(\boldsymbol{X})$

Return P_e and $\boldsymbol{X}$

End Procedure

Table 3 Pseudocode for the steepest local search procedure

Procedure Steepest_local_search(P_e,X)

//Record the currently best decision and its corresponding permutation: $\boldsymbol{X}^* = \boldsymbol{X}$, P_e^*=P_e

Repeat

Implement a selective 2-opt operation on P_e to get a new permutation $P_e^{'}$.

Apply the construction procedure to get a new decision $\boldsymbol{X}^{'}$ w.r.t. $P_e^{'}$.

If $J(\boldsymbol{X}^{'}) > J(\boldsymbol{X}^*)$ Then $\boldsymbol{X}^* = \boldsymbol{X}^{'}$, P_e^*=$P_e^{'}$ **End If**

Until $\forall\ P_e^{'} \in \mathcal{N}\ (P_e)$: $J(\boldsymbol{X}^{'}) \leqslant J(\boldsymbol{X}^*)$

Return X^* and P_e^*

End Procedure

exploration and local refinement[28], we employ truncated local searches. In the implementation of the selective 2-opt operations above, all UAAPs are sorted according to their values, and only 10% of all UAAPs with larger values will be considered for exchange.

We call the memetic algorithms corresponding to the greedy local search (GLS) and the steepest local search (SLS), MA-GLS and MA-SLS, respectively.

3.2 A genetic algorithm for DWTA

The design of genetic algorithms mainly encompasses the encoding of solutions (i.e., the representation of solutions), the initialization method, the design of evolutionary operators and the setting of parameters. Since a suitable representation has been proposed above for DWTA solutions, the following will focus on the latter three contents.

1) Initialization. The initial population is generated randomly without any heuristic information. This is mainly because effective constructive heuristics for DWTA may have a high complexity, and random starting solutions in combination with local refinement approaches, as shown above in the common procedure of MAs, can produce relatively good solutions.

2) Evolutionary operator 1 — Mutation. The function of mutation in GAs is to randomly generate a new solution from an individual in the current population. It is regarded as an important mechanism to keep population diversity. The mutation operation for DWTA is described as follows:

First, randomly select m UAAPs from S_{UAAP}. The number of selected UAAPs, i.e., m, is randomly selected from the set $\{1, 2, \ldots, \mathrm{size}\ (S_{\mathrm{UAAP}})\}$. Then, move the selected UAAPs in seq-

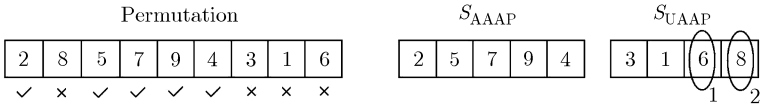

Figure 2 A mutation example.

uence to the head of the mutated permutation. An illustrative example for the mutation is shown in Figure 2.

3) Evolutionary operator 2 — Crossover. The main idea behind crossover is the inheritance of "good" genes from parents. As often as it is, two parent individuals (solutions) are randomly selected for each crossover operation to generate an offspring (new solution). Besides, a rule of thumb for crossover is to preserve the alleles that are identical for the same genes in both parents[26]. That is, it is checked whether each AAAP in the S_{AAAP} for one of the two parent permutations exists in the S_{AAAP} for the other parent. The common AAAPs in both parents will be inherited and placed at the head of their offspring. Particularly, the order of the common AAAPs in the offspring will be the same as that in the parent whose fitness (i.e., objective value) is better. The remaining AAPs for the offspring will be arranged randomly in its permutation.

4) Evolutionary operator 3 — Selection. The widely applied selection strategy, $(\mu+\lambda)$-selection, is adopted since it has been proved to have good performance in many researches[17,26,27].

5) Parameter settings. For GAs, the primary parameters include population size (PS), crossover probability (P_{c}) and mutation probability (P_{m}). However, in the context of MAs[26,27], the latter two parameters are not necessary. Instead, the number of crossover and mutation operations per generation is adopted. Considering the computational complexity of the local search approaches above, we follow the idea from Merz and Freisleben[26] to set $PS = 40$, and the number of crossovers and that of mutations per generation are both set at 10.

4 Computational experiments

4.1 Test-case generator

Given W, T, K and S, the generator will provide the essential parameters for a DWTA instance which include $\boldsymbol{V} = [v_k]_{1\times K}$, $\boldsymbol{Q} = [q_{jk}]_{T\times K}$, $\boldsymbol{P}^t = [p_{ij}(t)]_{W\times T}$, $N_i(i = 1, 2, \ldots, W)$ and $\boldsymbol{F}^t = [f_{ij}(t)]_{W\times T}(t = 1, 2, \ldots, S)$. The instructions for these parameters can be found in sections 2.1 and 2.2.

The values of assets, i.e., $v_k(k = 1, 2, \ldots, K)$, are randomly generated in the interval (10, 100). Since the number of targets is not less than that of threatened assets, that is $T \geqslant K$, we assume that the aim of the kth target is the kth asset $(k = 1, 2, \ldots, K)$ to ensure that each asset is threatened by at least one target. The aims of the remaining targets will be randomly selected from K assets. Any target has no threat to the assets which are not its aims. The parameters $q_{jk}(j = 1, 2, \ldots, T; k = 1, 2, \ldots, K)$ are randomly generated in the interval (0.6, 0.99). The parameters $p_{ij}(t)(i = 1, 2, \ldots, W; j = 1, 2, \ldots, T; t = 1, 2, \ldots, S)$ are randomly generated in the interval (0.4, 0.9). The engagement feasibility parameters $\boldsymbol{F}^t(t = 1, 2, \ldots, S)$ are also generated in a similar way. However, more zeros will appear in $\boldsymbol{F}^t$ corresponding to later stages, which accords with the fact that fewer weapons can be used in later sta-

ges. Note that $f_{ij}(t) = 0$ implies that weapon i cannot engage target j in stage t. The generation of $\boldsymbol{F}^t(t = 1, 2, \ldots, S)$ is shown as follows:

$$ratio(t) = f_L + (f_H - f_L)(t-1)/(S-1)$$

$$\text{for } t = 1, 2, \ldots, S;$$

$$f_{ij}(t) = [sign(rand - ratio(t)) + 1]/2$$

$$\text{for } i = 1, 2, \ldots, W; j = 1, 2, \ldots, T; t = 1, 2, \ldots, S,$$

where $ratio(t)$ is the probability that $f_{ij}(t)$ is set to 0, f_L and f_H are predefined constants with $0 < f_L < f_H < 1$ which reflect the lower and upper bounds of $ratio(t)$, respectively, $rand$ represents a random number in (0,1), and the function $sign(\cdot)$ is equal to 1 if its argument is positive, -1 otherwise. In our research, f_L and f_H are set at 0.1 and 0.9, respectively.

The generation of N_i $(i = 1, 2, \ldots, W)$ is categorized into three cases so as to cover different defense scenarios.

Case 1. One weapon, one shot. $N_i = 1$ for $i = 1, 2, \ldots, W$.

Case 2. No weapon can be used through all stages.

$N_i = \lceil S \cdot rand/2 \rceil$ for $i = 1, 2, \ldots, W$;

Case 3. All weapons are available in all stages. $N_i = S$ for $i = 1, 2, \ldots, W$.

Case 1 represents a scenario of "insufficient" ammunition since each weapon has at most one "bullet". In contrast, Case 3 corresponds to a scenario of "adequate" ammunition and each weapon can be used through all defense stages. Case 2 stands for a moderate one between the two extremes represented by Case 1 and Case 3. The three typical cases are considered for a comprehensive comparison of the DWTA algorithms under test. In our simulation, the number of stages (S) is fixed at 4. The setting of the other three primary parameters W, T and K includes the following cases: $W10T10K10$ (No. 1), $W100T50K50$ (No. 2), $W50T100K50$ (No. 3), $W100T100K50$ (No. 4). For practical military decision-making, the latter three cases with four defense stages, i.e., No. 2, No. 3 and No. 4, represent DWTA scenarios of large enough scale.

4.2 Experimental results

The DWTA decision-making algorithms under test include the GA, MA-GLS, MA-SLS proposed in section 3. The maximal number of function evaluations is set as $NFE_{\max} = 5W \cdot T \cdot S$. In other words, any algorithm will be terminated once $NFE_{\max}$ DWTA decisions have been analyzed. This setting is beneficial to observe the performance differences of the algorithms under test. Each algorithm was run 30 times for each test case with the results statistically analyzed. All algorithms were performed on a PC with Intel(R) 2.0 GHz CPU and 2.0 GB internal memory.

The statistical results are presented in Table 4. In columns 2–4 of Table 4, the data in the first and second rows for each case correspond to the mean and standard deviation of finally discovered best objective values (DBOV) and that of running time, respectively. The best result in each case is highlighted in bold. The results of two-tailed t-tests (significance level: 0.05) for paired comparisons between DWTA algorithms are also provided as shown in columns 5–7 of Table 4. A value of one indicates that the first algorithm for a paired comparison performs with 95% certainty better than the second algorithm. A value of -1 indicates that the second algorithm outperforms the first one in the aspect of optimization result or computation time. A value of zero implies that the performances are not statistically different. For example, in the case labeled as "No. 1, Case 1", MA-GLS (A2) is superior to GA (A1) in respect of DBOV with 95% certainty since the corresponding t-test result is one.

The convergence curves for the three algorithms in each case are shown in Figure 3. Note that these curves are the average of 30 runs. The three tested algorithms perform comparatively when solving the two simple instances of DWTA problem corresponding to "No. 1, Case 2" and "No. 1, Case 3". The numbers of both weapons and targets in the two instances are relatively small and the equipped ammunition is sufficient, especially for "No. 1, Case 3", to guarantee a strong defense, so it is easy for the three algorithms to find a desirable high-quality decision. As shown in Figure

Table 4 Statistical results about different algorithms

Case	Algorithms			t-test		
	A_1:GA	A_2:MA-GLS	A_3:MA-SLS	A_2 vs. A_1	A_2 vs. A_3	A_3 vs. A_1
No. 1	337.0±1.4	**345.2±2.1**	343.6±2.5	1	0	1
Case 1	**0.64±0.03(s)**	0.68±0.03(s)	0.70±0.03(s)	0	0	−1
No. 1	382.7±1.2	**383.1±2.8**	382.9±1.5	0	0	0
Case 2	**0.61±0.01(s)**	0.65±0.02(s)	0.68±0.02(s)	−1	0	−1
No. 1	389.3±1.0	**391.3±2.2**	389.5±2.3	0	0	0
Case 3	**0.63±0.01(s)**	0.68±0.02(s)	0.70±0.02(s)	−1	0	−1
No. 2	1707.3±4.8	**1743.7±5.2**	1721.8±6.7	1	1	1
Case 1	**118.2±0.2(s)**	120.3±0.4(s)	122.7±0.5(s)	−1	1	−1
No. 2	1739.4±4.4	**1753.5±5.1**	1744.2±6.3	1	1	0
Case 2	**116.3±0.1(s)**	118.1±0.3(s)	121.8±0.3(s)	−1	1	−1
No. 2	1751.8±4.2	**1777.2±4.9**	1753.3±6.3	1	1	0
Case 3	**112.7±0.1(s)**	117.6±0.3(s)	121.3±0.3(s)	−1	1	−1
No. 3	1441.6±6.2	**1522.1±6.3**	1467.5±7.2	1	1	1
Case 1	**124.4±0.2(s)**	128.2±0.4(s)	129.5±0.5(s)	−1	1	−1
No. 3	1463.2±5.5	**1541.9±6.1**	1491.2±6.9	1	1	1
Case 2	**120.1±0.1(s)**	124.3±0.2(s)	126.9±0.3(s)	−1	1	−1
No. 3	1471.8±5.3	**1548.3±5.9**	1527.1±6.7	1	1	1
Case 3	**121.3±0.1(s)**	123.7±0.2(s)	125.2±0.4(s)	−1	1	−1
No. 4	1492.3±4.8	**1583.1±5.2**	1547.4±5.8	1	1	1
Case 1	**213.8±0.3(s)**	220.6±0.4(s)	225.3±0.6(s)	−1	1	−1
No. 4	1537.5±4.2	**1615.3±5.0**	1587.6±5.3	1	1	1
Case 2	**210.2±0.2(s)**	217.2±0.3(s)	221.7±0.4(s)	−1	1	−1
No. 4	1547.2±4.1	**1621.7±4.9**	1591.3±5.3	1	1	1
Case 3	**210.3±0.2(s)**	218.5±0.3(s)	222.5±0.4(s)	−1	1	−1

3, the performance differences between the three algorithms increase with the scale of DWTA problems. MA-GLS prominently outperforms GA and MA-SLS in solving large-scale DWTA instances from No. 2 through No. 3 to No. 4. MA-SLS also has obvious advantages over GA in all large-scale cases except "No. 2, Case 3". The results shown in Table 4 also indicate that the algorithm MA-GLS has obvious advantages over GA and MA-SLS especially in the decision-making of large-scale DWTA problems. It should be noted that MA-GLS and MA-SLS are a little more time-consuming than GA in all cases. However, the trivial advantage of GA in computation time, as shown in Table 4, may be neglected in contrast to its optimization quality. Compared with GA, MA-SLS performs better in the aspect of optimization quality. The advantage of MA-GLS and MA-SLS over GA dem-

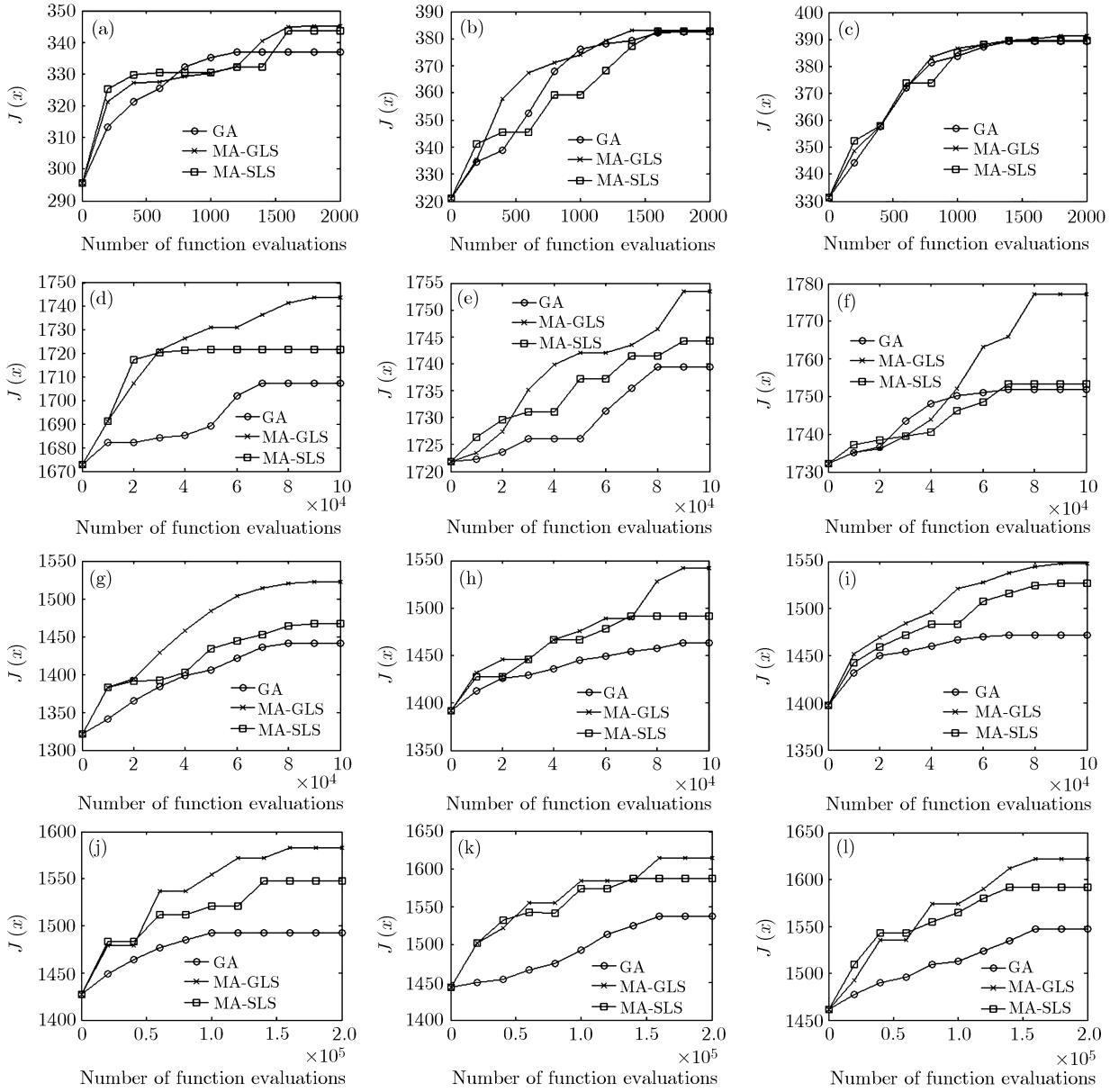

Figure 3 Convergence curves for GA, MA-GLS and MA-SLS. (a) No. 1, Case 1; (b) No. 1, Case 2; (c) No. 1, Case 3; (d) No. 2, Case 1; (e) No. 2, Case 2; (f) No. 2, Case 3; (g) No. 3, Case 1; (h) No. 3, Case 2; (i) No. 3, Case 3; (j) No. 4, Case 1; (k) No. 4, Case 2; (l) No. 4, Case 3.

onstrates that the cooperation of local search methods and global search implemented by GA can improve the quality of DWTA decision-making greatly. The advantage of MA-GLS over MA-SLS shows that the utilization of greedy local search, in contrast to steepest local search, can achieve better tradeoff between the exploration and exploitation of DWTA decision space.

5 Conclusions

In order to solve asset-based DWTA problems with complicated constraints, a general "virtual" representation of decisions which is in essence the permutation of all assignment pairs was proposed to facilitate the generation of feasible decisions. A construction procedure which operates on the virtual representation can guarantee the satisfac-

tion of all constraints. Based on the construction technique, three evolutionary decision-making algorithms including a genetic algorithm and two memetic algorithms were proposed to solve DWTA problems. The proposed memetic algorithm based on greedy local search can generate obviously better DWTA decisions without the cost of overmuch extra computation time than genetic algorithm and the memetic algorithm based on steepest local search.

The authors would like to thank the anonymous reviewers for their constructive comments on this paper

1 Athans M. Command and control (C2) theory: a challenge to control science. IEEE Trans Automat Contr, 1987, 32(4): 286–293

2 Lloyd S P, Witsenhausen H S. Weapon allocation is NP-complete. In: Proceedings of the IEEE Summer Computer Simulation Conference, Nevada, USA, 1986. 1054–1058

3 Hosein P A, Athans M. Preferential defense strategies. Part I: The static case. Report LIPS-P-2002. 1990

4 Hosein P A, Athans M. Preferential defense strategies. Part II: The dynamic case. Report LIPS-P-2003. 1990

5 Li J, Cong R, Xiong J. Dynamic WTA optimization model of air defense operation of warships' formation. J Syst Eng Electr, 2006, 7(1): 126–131

6 Cai H, Liu J, Chen Y, et al. Survey of the research on dynamic weapon-target assignment problem. J Syst Eng Electr, 2006, 17(3): 559–565

7 Malhotra A, Jain R K. Genetic algorithm for optimal weapon allocation in multilayer defence scenario. Defence Sci J, 2001, 51(3): 285–293

8 Bisht S. Hybrid genetic-simulated annealing algorithm for optimal weapon allocation in multilayer defence scenario. Defence Sci J, 2004, 54(3): 395–405

9 Karasakal O. Air defense missile-target allocation models for a naval task group. Comput Oper Res, 2008, 35: 1759–1770

10 Wacholder E. A neural network-based optimization algorithm for the static weapon-target assignment problem. ORSA J Comput, 1989, 1(4): 232–246

11 Grant K E. Optimal resource allocation using genetic algorithms. Naval Review, Naval Research Laboratory, Washington, DC, 1993. 174–175

12 Lu H, Zhang H, Zhang X, et al. An improved genetic algorithm for target assignment optimization of naval fleet air defense. In: Proceedings of the 6th World Congress on Intelligent Control & Automation, Dalian, China, 2006. 3401–3405

13 Cullenbine A C. A taboo search approach to the weapon assignment model. Master Thesis, Department of Operational Sciences, Air Force Institute of Technology, 2000

14 Li H R, Miao Y. WTA with the maximum kill probability based on simulated annealing algorithms. In: Proceedings of the Conference on Special Committee of C2 and Computer of the Electronic Technology Academic Committee of China, Ship Engineering Society, 2000. 436–440

15 Lee Z J, Lee C Y, Su S F. An immunity-based ant colony optimization algorithm for solving weapon-target assignment problem. Appl Soft Comput, 2002, 2: 39–47

16 Zeng X, Zhu Y, Nan L, et al. Solving weapon-target assignment problem using discrete particle swarm optimization. In: Proceedings of the 6th World Congress on Intelligent Control & Automation, Dalian, China, 2006. 3562–3565

17 Lee Z J, Su S F, Lee C Y. Efficiently solving general weapon-target assignment problem by genetic algorithms with greedy eugenics. IEEE Trans Syst Man Cybern-B, 2003, 33(1): 113–121

18 Fu T, Liu Y, Chen J. Improved genetic & ant colony optimization for regional air defense WTA problem. In: Proceedings of the 1st International Conference on Innovative Computing, Information and Control (ICICIC'06), Dalian, China, 2006. 226–229

19 Ahuja R K, Kumar A, Jha K C, et al. Exact and heuristic algorithms for the weapon-target assignment problem. Oper Res, 2007, 55(6): 1136–1146

20 Kwon O, Lee K, Kang D, et al. A branch-and-price algorithm for a targeting problem. Naval Res Logist, 2007, 54: 732–741

21 Hosein P A, Walton J T, Athans M. Dynamic weapon-target assignment problems with vulnerable C2 nodes. Report LIDS-P-1786. 1988

22 Hosein P A, Athans M. Some analytical results for the dynamic weapon-target allocation problem. Report LIDS-P-1944. 1990

23 Khosla D. Hybrid genetic approach for the dynamic weapon-target allocation problem. In: Proceedings of SPIE, 2001, 4396: 244–259

24 Havens M E. Dynamic allocation of fires and sensors. Master Thesis, Naval Postgraduate School, OMB No. 0704-0188, 2002

25 Loiola E M, Maia de Abreu N M, Boaventura Netto P O, et al. A survey for the quadratic assignment problem. Euro J Oper Res, 2007, 176: 657–690

26 Merz P, Freisleben B. Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. IEEE Trans Evol Comput, 2000, 4(4): 337–352

27 Krasnogor N, Smith J. A tutorial for competent memetic algorithms: model, taxonomy, and design issues. IEEE Trans Evol Comput, 2005, 9(5): 474–488

28 Chen J, Xin B, Peng Z H, et al. Optimal contraction theorem for exploration-exploitation tradeoff in search and optimization. IEEE Trans Syst Man Cybern-A, 2009, 39(3): 680–691