An Estimation of Distribution Algorithm with Efficient Constructive Repair/Improvement Operator for the Dynamic Weapon-Target Assignment

XIN Bin[1,2], CHEN Jie[1,3]

1. School of Automation, Beijing Institute of Technology, Beijing 100081
E-mail: brucebin@bit.edu.cn

2. Decision and Cognitive Sciences Research Centre, Manchester Business School, University of Manchester, Manchester M15 6PB

3. Key Laboratory of Complex System Intelligent Control and Decision, Ministry of Education, Beijing 100081
E-mail: chenjie@bit.edu.cn

Abstract: This paper reports our recent research about new efficient problem-solvers for the dynamic weapon-target assignment (DWTA). A binary-encoding-based estimation of distribution algorithm (EDA) is proposed to solve DWTA problems. An elaborate constructive repair/improvement (CRI) operator is proposed and integrated into the EDA to achieve constraint saturation, which conduces to constraint satisfaction as well as the improvement of generated solutions. The performance comparison against another two EDAs which employ well-known constraint handling methods demonstrates the superiority of the CRI operator. The proposed EDA based on the CRI operator also shows very competitive and even better performance against several state-of-the-art DWTA algorithms.

Key Words: Dynamic weapon-target assignment, estimation of distribution algorithm, constructive repair/improvement operator, constraint handling

1 Introduction

The dynamic weapon-target assignment (DWTA) problem is a typical NP-Complete combinatorial optimization problem which is rooted in military operations research[1]-[4]. In practice, commanders have to react to the battle situation quickly while deliberating the whole combating effect through all stages to ensure the quality of decision schemes on resource allocation. The objective of DWTA is to minimize the own-force damage in defense scenarios or maximize the damage of enemies in offense scenarios by assigning available weapons to hostile targets on appropriate engagement occasions. In contrast to the static weapon-target assignment (SWTA) in which all weapons are assigned to targets in a single stage[1], a distinct feature of DWTA lies in that DWTA considers the pairing of weapons and targets among multiple engagement stages[2]. In other words, the resource allocation in DWTA involves the triplet pairing of weapons, targets and stages[5]-[7]. In the following, we provide a brief review on previous DWTA research.

Most previous research on the weapon-target assignment has been focused on SWTA[4]. DWTA began to gain more attention of researchers only in recent years though it was formally put forward by Hosein and Athans in 1990[2]. Cai et al. provided a survey on WTA research and set forth some basic concepts on DWTA[4]. Hosein and Athans made a pioneering effort to analyze a two-stage DWTA process and proposed a suboptimal algorithm to find some desirable solutions[8]. Khosla proposed a hybrid algorithm of genetic algorithm and simulated annealing to solve a DWTA problem for network-centric force coordination[9]. Zacherl designed a genetic algorithm to solve the DWTA problem involved in a UAV's mission of striking time-sensitive targets[10]. Wu et al. proposed a DWTA algorithm based on genetic algorithm which handles targets one by one according to the deadlines of each weapon-target assignment pair[11]. Dionne et al. developed a sequential DWTA algorithm for naval warfare which considers all potential decisions and suffers from dimensional explosion[12]. Li et al. proposed a target-based DWTA model which aims at minimizing the total threat of the targets surviving through the whole defense process[13]. Recently, we built a generic defense-oriented asset-based DWTA model which incorporates different kinds of practical constraints[5]. We proposed an efficient permutation-based tabu search algorithm to solve the DWTA problem[6]. The tabu search algorithm takes advantage of a construction procedure to convert the permutations of available weapon-target-stage assignment pairs into saturated feasible solutions. In order to fit the real-time requirement of DWTA, we also proposed a constructive heuristic which uses the domain knowledge of DWTA in the form of three simple rules to obtain a high-quality solution very quickly[7]. The heuristic has a notable virtue of low computational complexity.

The contribution of this paper is twofold. First, a new sophisticated constructive repair/improvement operator is proposed to handle constraints in DWTA. The operator can be applied in any binary-encoding based DWTA algorithms. Second, an estimation of distribution algorithm is proposed to solve DWTA problems more efficiently.

2 Mathematical Model of the DWTA Problem

The objective of the DWTA decision-making process is to maximize the expected total value of own-force assets surviving through the whole defense process:

*This work is supported by National Science Fund for Distinguished Young Scholars under Grant 60925011.

$$J(\mathbf{X}) = \sum_{k=1}^{K} v_k \prod_{j \in \mathscr{T}_k} \left[1 - q_{jk} \prod_{t=1}^{S} \prod_{i \in \mathscr{W}_j^t} (1 - p_{ij}(t))^{x_{ij}(t)} \right] \quad (1)$$

where K and S are the number of threatened assets and that of defense stages, respectively, t is the stage index, $\mathbf{X} = [X_1, X_2, \cdots, X_S]$ with $X_t = [x_{ij}(t)]_{W \times T}$ is the decision matrix at stage t, and $x_{ij}(t) = 1$ if weapon i is assigned to target j at stage t, $x_{ij}(t) = 0$ otherwise; $\mathscr{T}_k$ is the index set of the targets which threaten asset k; $\mathscr{W}_j^t$ is the index set of the weapons which are assigned in stage t to intercept target j; v_k is the value of asset k; q_{jk} is the lethality probability that target j destroys asset k, and $p_{ij}(t)$ is the lethality probability that weapon i destroys target j at stage t which can be evaluated according to actual combat situations.

The following four kinds of constraints are considered in the DWTA model[5]-[7]:

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

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

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

$$x_{ij}(t) \le f_{ij}(t), \quad \forall t \in \{1,2,\cdots,S\}, \ \forall i \in \{1,2,\cdots,W\}, \quad \forall j \in \{1,2,\cdots,T\} \quad (5)$$

where T and W are the number of targets and that of weapons, respectively.

The constraint (2) reflects the weapons' capability of striking multiple targets at the same time. In practice, most weapons can only engage a single target each time. Besides, a superior weapon capable of engaging multiple targets concurrently can be regarded as multiple separate weapons. Accordingly, we set $n_i = 1$ for $\forall i \in \{1,2,\cdots,W\}$. The constraint (3) restricts the ammunition consumption for each target at each stage. We assume that $m_j = 1$ for $\forall j \in \{1,2,\cdots,T\}$. This is a rational setting for combating systems with high accuracy and lethality, consistent with the common "shoot-look-shoot" engagement policy[2]. The constraint (4) is a resource constraint, reflecting the amount of ammunition available for weapons. $N_i \ (i = 1,2,\cdots,W)$ is the maximal number of times that weapon i can be used. The constraint (5) reflects the engagement feasibility with $f_{ij}(t)$ indicating whether weapon i can be used to strike target j at stage t. $f_{ij}(t) = 0$ if weapon i cannot shoot target j at stage t for any potential reason (e.g., the time window of targets and weapons[4],[11],[13]), and $f_{ij}(t) = 1$ otherwise.

A complete DWTA process is a multi-stage decision-making process. At each decision-making stage, the decision-maker (DM) needs to work out a global assignment scheme which takes into account the whole defense effect through all subsequent defense stages[6]. However, only the assignments which can be implemented immediately at the current stage are carried out. When entering the next stage, the DM has observed the outcomes of previous engagements and has to reformulate a new global assignment scheme which covers the subsequent stages starting from the next stage[7]. Nevertheless, from the perspective of DWTA problem-solving, the DM at each stage faces a similar global assignment problem. Besides, the DWTA problems corresponding to latter stages usually become much easier due to the reduction of targets, available weapons and engagement occasions. Therefore, the same DWTA algorithm can be applied at different stages, which means that one is only interested in obtaining assignments for the present stage[2]. In the following, we will focus on the DWTA problem-solving at one decision-making stage to design and evaluate DWTA algorithms.

3 Estimation of Distribution Algorithm for DWTA problem-solving

Estimation of distribution algorithms (EDAs), also termed as probabilistic model-building genetic algorithms[14]-[15], are population-based stochastic search algorithms that explore the space of potential solutions by building and sampling the probability model of promising candidate solutions. EDAs maintain a population of solutions and follow a cyclic operation procedure composed by three primary operations: choosing a subset of elite solutions (selection), building a probability model from the chosen solutions (modeling), and generate new solutions from the constructed model (sampling).

3.1 Solution Encoding and Probability Model

An appropriate solution encoding scheme is crucial for any competent algorithm to solve combinatorial optimization problems. The decision matrix $\mathbf{X}$, apparently being a straightforward representation of solutions, is not an efficient encoding scheme as it contains many decision variables ($x_{ij}(t)$) which may violate the constraints in (5). In fact, it is unnecessary to include the variables which correspond to $f_{ij}(t) = 0$ since $x_{ij}(t) = 1$ will obviously violate constraints in this case. Therefore, we only consider feasible assignment pairs corresponding to $f_{ij}(t) = 1$ which are also termed as available assignment pairs (AAPs) in previous research[5]-[7]. Arrange all AAPs by $AAP_1, AAP_2, \cdots, AAP_L$ where L denotes the number of AAPs and $AAP_k = (i_k, j_k, t_k)$ ($k = 1,2,\cdots,L$) is a triplet pairing of weapon i_k, target j_k and stage t_k. We employ an L-sized 0-1 vector $\mathbf{z} = [z_1, z_2, \cdots, z_L]$ to represent a DWTA solution where $z_k \ (k = 1,2,\cdots,L)$ indicates whether AAP_k is chosen ($z_k = 1$, equivalently, $x_{i_k j_k}(t_k) = 1$) or not ($z_k = 0$, i.e., $x_{i_k j_k}(t_k) = 0$).

The EDA's probability model for DWTA problem-solving is shown as follows:

$$p(z_k = 1) = \min\left\{ \max\left\{ \frac{1}{N} \sum_{n=1}^{N} z_k^n, 0.01 \right\}, 0.99 \right\}, \quad p(z_k = 0) = 1 - p(z_k = 1), \ k = 1,2,\cdots,L. \quad (6)$$

where $p(z_k = 1)$ is the probability that the component z_k of elite solutions is one; $\mathbf{z}^n = [z_1^n, z_2^n, \cdots, z_L^n] \ (n = 1,2,\cdots,N)$

are elite solutions chosen for modeling; and N is the number of elite solutions, i.e., the sample size. 0.01 and 0.99 are the lower and upper bounds for $p(z_k = 1)$ to avoid the premature convergence of the probability model.

The rule for generating new solutions from the above probability model is shown as follows:

$$z_k = \begin{cases} 1, & \text{if } rand < p(z_k = 1), \\ 0, & \text{otherwise.} \end{cases} \text{ for } k = 1, 2, ..., L \quad (7)$$

where $rand$ is a random number generated from the interval (0,1).

Remark 1. The simple probability model was also employed in the classical univariate marginal distribution algorithm[16]. The linkage between different solution components may be considered to build a more sophisticated probability model such as a tree-based model or Bayesian network[14]-[15]. However, complicated models may cause a serious issue of computational complexity which is unacceptable for real-time DWTA decision-making.

3.2 Constructive Repair/Improvement Operator

The solutions which will be generated from the above model may not satisfy the constraints in (2), (3) or (4), resulting in infeasible solutions. Besides, even if feasible solutions can be produced, they may be unsaturated, meaning that more AAPs can be chosen to produce better solutions without any constraint violations. It should be noted that a saturated feasible solution contains as many assigned AAPs as possible, and any further addition of other AAPs will give rise to constraint violation. From the above considerations, in the following we propose a novel constructive repair/improvement operator to ensure constraint satisfaction as well as improve the quality of feasible solutions:

Step 1. Convert the solution $\mathbf{z} = [z_1, z_2, \cdots, z_L]$ generated by the probability model into a new vector $\mathbf{r} = [r_1, r_2, \cdots, r_L]$ as follows:

$$r_k = z_k * rand_k^1 - (1 - z_k) * rand_k^2, \text{ for } k = 1, 2, ..., L \quad (8)$$

where $rand_k^1$ and $rand_k^2$ are two random numbers generated from the interval (0,1).

Step 2. Sort all AAPs in the descending order of the values of their corresponding elements in the vector $\mathbf{r}$. For example, the vector $\mathbf{r} = [-0.1, 0.3, 0.8, -0.6, 0.2]$ which is generated from $\mathbf{z} = [0,1,1,0,1]$ will give the sorting result: AAP_3, AAP_2, AAP_5, AAP_1, AAP_4.

Step 3. Use the following construction procedure to generate a saturated feasible solution:

For i = 1 to L

> Check if assigning the ith AAP ($AAP_{k(i)}$) in the sorted sequence will violate any constraints: if no constraints are violated, let $z_{k(i)} = 1$; otherwise, let $z_{k(i)} = 0$.

End

The above constructive repair/improvement (CRI) operator has the following features:

1) Be able to retain the information contained in the probability model to large extent. On one hand, in the solution generated from the probability model, the chosen AAPs will be endowed with positive *r* values, and the unchosen ones lead to negative *r* values. Consequently, the chosen AAPs will be arranged to the head of the AAP sequence in Step 2, and they will be preferred to be rechosen in Step 3. On the other hand, the chosen AAPs which do not violate constraints will be reserved in the solution finally produced by Step 3.

2) The construction procedure in Step 3 guarantees that all solutions generated by the CRI operator are saturated feasible solutions. The operator can not only repair infeasible solutions but also improve the quality of feasible solutions.

3) Among all the chosen AAPs generated by EDA's sampling procedure, the randomizing operation shown in (8) shows no preference to any of them. Therefore, any AAP which may have conflicts against other AAPs will have an opportunity to be included in the final solution produced by the CRI operator.

3.3 EDA

The procedure of the proposed EDA is presented as follows:

Step 1. *Initialization*

Randomly generate $PS - 1$ solutions with $p(z_k = 1) = 0.5$ ($k = 1, 2, \cdots, L$) and use the CRI operator to repair and improve the initial solutions. Use the rule-based heuristic proposed in [7] to generate a solution and add it into the population. Record the best solution in the whole population.

Remark 2. The rule-based heuristic in [7] is *de facto* a deterministic greedy algorithm which chooses AAPs according to their $v_k q_{jk} p_{ij}(t)$ values.

Step 2. *Elitist Selection*

Sort the PS solutions in the descending order of their objective values. Select the $N = PS * R$ ($R \in (0,1)$) top solutions to build the probability model [see (6)].

Step 3. *Sampling*

Use the constructed model to generate $PS - N$ new solutions, and use the CRI operator to repair and improve them. Replace the worst $PS - N$ solutions in the current population by the new solutions. Update the so-far-best solution.

Step 4. *Termination Criterion*

If the termination criterion is satisfied, terminate the algorithm and output the so-far-best solution; otherwise, go to Step 2.

4 Computational Experiments and Performance Comparison

There are ten DWTA test instances involved in the experiments, including two simple small-sized instances in [6] and eight instances generated by the test-case generator proposed in [7]. The characteristic parameters of these instances are listed as follows.

Instance 1: *Naval Single-Platform Combat Scenario* (W5T3S3K1, L = 29)
Instance 2: *Ground-based Air Defense Scenario* (W8T5S3K5, L = 36)
Instance 3: W10T10S4K5 (L = 209)
Instance 4: W20T10S4K5 (L = 387)
Instance 5: W20T20S4K10 (L = 815)
Instance 6: W20T20S4K20 (L = 803)
Instance 7: W50T50S4K20 (L = 495)
Instance 8: W50T50S4K20 (L = 2023)
Instance 9: W100T50S4K50 (L = 1017)
Instance 10: W100T100S4K50 (L = 1999)

All experiments were carried out in Matlab (Version 6.5) on a laptop with Intel (R) Core i5 CPU (2.27GHz) and 1.92GB internal memory. Regarding any instance, all algorithms independently run 30 times and results are statistically analyzed. Without specific claims, any algorithm will be terminated when the number of function evaluations reaches $\min\{5W*T*S, 50000\}$. The EDA algorithm has two parameters: population size (PS) and selection ratio (R). Four kinds of settings are tested for each of them: $PS = 50, 100, 200, 500$ and $R = 0.1, 0.3, 0.5, 0.7$. There are totally sixteen combinations of the two parameters. It was found in preliminary experiments that $R = 0.3$ is always a nice choice, and a smaller population size ($PS = 50$) fits small-sized instances while a larger population size ($PS = 200$) is a better choice for large-sized instances. To save space, the detailed results are not shown here and the identified setting is described as follows:

$PS = 50$ and $R = 0.3$ for instances with $L \le 1000$; $PS = 200$ and $R = 0.3$ for instances with $L > 1000$.

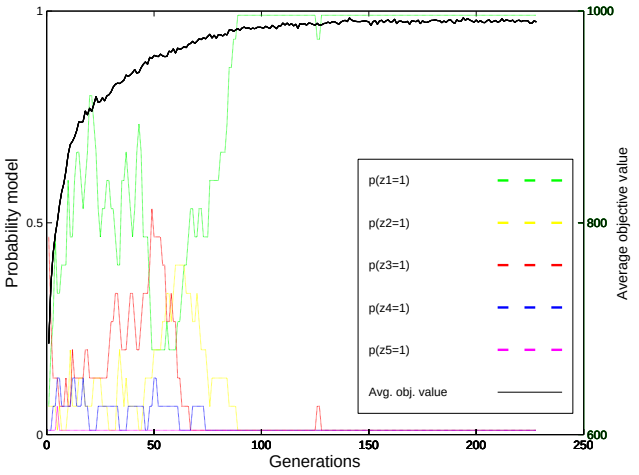

Fig. 1: The dynamic change of EDA's probability model and the EDA's convergence process (instance 7).

Fig.1 shows the dynamic change of the EDA's probability model and the convergence process of EDA in solving instance 7. For clarity, only five components of the probability model, $p(z_k = 1)$ ($k = 1, 2, 3, 4, 5$), are shown in Fig.1. It is clear that with the increase of generations, the probability model tends to polarization, that is to say, the probability values of the EDA model reach either its lower bound (0.01) or upper bound (0.99). Along with the polarization of the probability model, the algorithm converges as indicated by the curve of the average objective value of all solutions in the population.

The following experiments are arranged into two parts. The first part provides a comparison between different constraint handling methods which are embedded into the same EDA algorithm, including the CRI operator and two well-known constraint handling methods in the constrained evolutionary optimization research. The second part gives a comparison between the proposed EDA and two state-of-the-art DWTA algorithms.

4.1 Comparison on Constraint Handling Methods

There are many choices to handle constraints in constrained optimization[17]. The most common constraint handling methods are penalty functions which integrate the constraint handling and the improvement of objective values into a penalty function. Infeasible solutions will be penalized by decreasing (penalizing) their fitness according to their constraint violations. As pointed out by Runarrson and Yao, a crucial issue in the design of penalty functions is how to strike a balance between objective improvement and constraint satisfaction[18]. Runarsson and Yao proposed a stochastic ranking (SR) approach to balance the two goals in constrained optimization[18]. Deb proposed a straightforward approach based on three feasibility rules (FR)[19]. Both SR and FR allow the evolutionary population to include infeasible solutions. In SR, two feasible solutions are compared according to their objective values. In other cases, any two solutions are compared according to their objective values with probability P_f and according to their constraint violations with probability $1 - P_f$ [19]. Obviously, SR balances constraint satisfaction and objective improvement stochastically. In contrast, FR adopts three simple rules for pairwise comparison of solutions: 1) any feasible solution is preferred to any infeasible one; 2) among two feasible solutions, the one having better objective value is preferred; 3) among two infeasible solutions, the one having smaller constraint violation is preferred[19].

In the following, SR and FR will be incorporated into the EDA framework in the previous section to solve DWTA problems. For both SR and FR, the constraint violation of a solution (denoted by ϕ) is measured as follows:

$$\phi = \sum_{t=1}^{S}\sum_{i=1}^{W} H\left(\sum_{j=1}^{T} x_{ij}(t) - n_i\right) + \sum_{t=1}^{S}\sum_{j=1}^{T} H\left(\sum_{i=1}^{W} x_{ij}(t) - m_j\right) + \sum_{i=1}^{W} H\left(\sum_{t=1}^{S}\sum_{j=1}^{T} x_{ij}(t) - N_i\right)$$

where $H(\cdot)$ is equal to the value of its argument if the argument is positive, and zero otherwise.

For brevity, the EDAs based on SR, FR and CRI are termed as EDA-SR, EDA-FR and EDA-CRI, respectively. The three EDA variants adopt the same probability model, the same sampling method as well as the same setting of EDA parameters. However, EDA-SR and EDA-FR do not employ the CRI operator to repair and/or improve solutions.

Table 1: Statistical Results on Different DWTA Algorithms in the Form of the Mean plus Standard Deviation of Finally Discovered Best Objective Values in 30 Runs.

Instance No.	EDA-CRI	EDA-SR	EDA-FR	Tabu search[6]	Constructive heuristic[7]
1	0.53007 ± 0.00185	0.45629 ± 0.06675$^{(30)*}$	0.41763 ± 0.09191$^{(30)*}$	**0.53089 ± 0.00182**$^{\#}$	0.52763 ± 0*
2	171.104 ± 0	168.790 ± 1.579$^{(30)*}$	168.040 ± 1.905$^{(30)*}$	**171.104 ± 0**$^{\#}$	169.500 ± 0*
3	**312.76 ± 1.08**	275.21 ± 12.46$^{(30)*}$	273.72 ± 10.58$^{(30)*}$	309.41 ± 1.80*	309.41 ± 0.00*
4	**332.26 ± 0.08**	307.57 ± 6.78$^{(30)*}$	304.53 ± 7.74$^{(30)*}$	332.13 ± 0.10*	331.7 ± 0*
5	**577.73 ± 0.00**	357.75 ± 29.34$^{(29)*}$	325.29 ± 31.14$^{(28)*}$	560.20 ± 4.24*	**577.73 ± 0**$^{\#}$
6	**1296.7 ± 0.0**	1029.6 ± 37.7$^{(26)*}$	991.2 ± 39.1$^{(30)*}$	1283.5 ± 4.6*	**1296.7 ± 0**$^{\#}$
7	**1004.2 ± 1.8**	885.13 ± 29.23$^{(30)*}$	830.43 ± 21.51$^{(30)*}$	986.80 ± 6.32*	970.59 ± 0*
8	**928.42 ± 0.00**	429.38 ± 39.22$^{(22)*}$	410.50 ± 37.98$^{(25)*}$	912.05 ± 2.98*	**928.42 ± 0**$^{\#}$
9	**2988.7 ± 0.4**	2569.9 ± 37.8$^{(30)*}$	2514.5 ± 44.6$^{(30)*}$	2986.1 ± 0.8*	2980.9 ± 0*
10	**2464.1 ± 0.0**	1165.4 ± 74.1$^{(24)*}$	1081.5 ± 67.8$^{(24)*}$	2423.2 ± 7.6*	**2464.1 ± 0**$^{\#}$

The numbers in parentheses for EDA-SR and EDA-FR are the times that they find feasible solutions in each case.
* EDA-CRI performs better than the target algorithm, which is tested by the *Wilcoxon* rank sum test with a significance level 0.05.
$^{\#}$ The performances of EDA-CRI and the target algorithm have no significant difference.

Table 2: Statistical Results on Different DWTA Algorithms in the Form of the Mean plus Standard Deviation of the Computation Time (in seconds) in 30 Runs

Instance No.	EDA-CRI	EDA-SR	EDA-FR	Tabu search[6]	Constructive heuristic[7]
1	0.053 ± 0.008	0.071 ± 0.008	0.071 ± 0.008	0.035 ± 0.013	**0.001 ± 0.004**
2	0.160 ± 0.077	0.234 ± 0.000	0.234 ± 0.003	0.092 ± 0.027	**0.001 ± 0.004**
3	0.781 ± 0.036	0.823 ± 0.007	0.824 ± 0.007	0.458 ± 0.086	**0.003 ± 0.008**
4	2.090 ± 0.057	2.290 ± 0.008	2.290 ± 0.008	1.306 ± 0.180	**0.005 ± 0.007**
5	7.299 ± 0.223	7.447 ± 0.007	7.446 ± 0.008	8.213 ± 1.655	**0.014 ± 0.009**
6	7.134 ± 0.078	7.348 ± 0.007	7.347 ± 0.006	7.631 ± 1.515	**0.014 ± 0.009**
7	44.54 ± 1.42	48.43 ± 0.01	48.43 ± 0.01	20.29 ± 0.41	**0.011 ± 0.008**
8	108.99 ± 3.38	126.28 ± 0.00	126.28 ± 0.00	42.22 ± 1.28	**0.056 ± 0.019**
9	75.84 ± 0.64	78.17 ± 0.00	78.17 ± 0.00	33.35 ± 0.64	**0.036 ± 0.010**
10	129.84 ± 2.26	135.20 ± 0.00	135.20 ± 0.00	56.85 ± 1.65	**0.092 ± 0.026**

Besides, it was observed that EDA-SR and EDA-FR spend less time generating new solutions. In the experiments, EDA-SR and EDA-FR are allowed to run until the maximal time that EDA-CRI takes among 30 runs has been exceeded. In EDA-SR, three different settings for its parameter P_f are considered: $P_f = 0.1, 0.3, 0.45$ where $P_f = 0.45$ is suggested by Runarsson and Yao[18]. Only the results corresponding to the best setting are presented for EDA-SR in each case. The experimental results about the three EDAs are shown in Tables 1 and 2.

From Tables 1 and 2, it is obvious that EDA-CRI outperforms both EDA-SR and EDA-FR in all cases. EDA-SR and EDA-FR even failed to find feasible solutions in some cases. For example, EDA-SR and EDA-FR did not produce feasible solutions once and twice among 30 runs regarding instance 5, respectively. In fact, both EDA-SR and EDA-FR spend too much time on infeasible solutions. In contrast, EDA-CRI only produces saturated feasible solutions, which benefits EDA-CRI to focus its search on promising solutions.

4.2 Comparison against State-of-the-Art DWTA Algorithms

The tabu search (TS) based DWTA algorithm proposed in [6] and the rule-based constructive heuristic (CH) proposed in [7] are chosen for a further comparison with EDA-CRI. Like EDA, TS is a search algorithm; however, it relies on neighborhood exploration and diversification mechanisms to carry out a trajectory search rather than the population-based search in EDA. In contrast, CH is a deterministic greedy algorithm which produces a single solution without any iterative search. As claimed in [7], the main advantage of CH is its lower computational complexity which fits the DWTA requirement on real-time computation very well. Both TS and CH are employed to solve the above ten DWTA instances. The termination criterion for TS is same with that for EDA-CRI. The computational results are shown in Tables 1 and 2.

As shown in Table 1, EDA-CRI performs comparatively against TS in solving instances 1 and 2, and outperforms TS in all of the remaining cases. Compared with CH, EDA-CRI produces significantly better solutions in solving instances 1,

2, 3, 4, 7 and 9. It is not surprising that EDA-CRI did not lose to CH since EDA-CRI uses CH to provide an initial solution. However, EDA-CRI did not further improve the solution generated by CH in solving instances 5, 6, 8 and 10, which demonstrates that CH did provide high-quality solutions. From Table 2, there is no doubt that CH is the best DWTA algorithm in terms of time cost. This is because CH only relies on simple rules to generate a single solution without any iterative search. However, it is noteworthy that if the time for DWTA decision-making permits, any effort to improve the quality of decision schemes should be preferred. As shown in Table 2, EDA-CRI can solve smaller-sized instances (instances 1-6) within a few seconds, and solve larger-sized ones (instances 7-10) within a few minutes. In fact, the time cost can be reduced since EDA-CRI has converged earlier before the termination criterion is satisfied (see Fig.1). Besides, since EDA-CRI is a population-based optimizer, it is easy to implement EDA-CRI in parallel if the computing platform supports parallel computation, which can further reduce its time cost. In this sense, EDA-CRI is a desirable choice for DWTA problem-solving.

5 Conclusion and Future Work

An estimation of distribution algorithm based on a novel constructive repair/improvement operator is proposed for DWTA problem-solving. The repair/improvement operator not only effectively exploits the information on promising solutions extracted by EDA's probability model, but also helps to achieve constraint satisfaction as well as improve the quality of feasible solutions. It was shown that the repair/improvement operator is superior to two well-known constraint handling methods in solving DWTA problems. The proposed EDA incorporates the sophisticated operator and uses a rule-based constructive heuristic to provide an initial solution. Experimental results demonstrate the efficiency of the proposed EDA in DWTA problem-solving.

As a promising research line, it is possible to incorporate the domain knowledge of DWTA (e.g., rules similar to those used by the constructive heuristic in [7]) into EDA more efficiently, especially in the process of building EDA's probability model. The search bias induced by appropriate heuristic rules may accelerate the convergence towards high-quality DWTA solutions and even global optimal decision schemes. How to build a more competent probability model for EDA deserves further research.

References

[1] P. A. Hosein and M. Athans, Preferential defense strategies-Part I: The static case, MIT Laboratory for Information and Decision Systems with partial support, USA, Tech. Rep. LIPS-P-2002, 1990.

[2] P. A. Hosein and M. Athans, Preferential defense strategies-Part II: The dynamic case, MIT Laboratory for Information and Decision Systems with partial support, USA, Tech. Rep. LIPS-P-2003, 1990.

[3] S. P. Lloyd and H. S. Witsenhausen, Weapon allocation is NP-complete, in *Proc. IEEE Summer Simulation Conference*, Reno, Nevada, USA, 1986: 1054–1058.

[4] H. Cai, J. Liu, Y. Chen, and H. Wang, Survey of the research on dynamic weapon-target assignment problem, *J. Syst. Eng. Elec.*, 17(3): 559–565, 2006.

[5] J. Chen, B. Xin, Z. H. Peng, L. H. Dou, and J. Zhang, Evolutionary decision-makings for dynamic weapon-target assignment problem, *Science in China Series F: Information Sciences*, 52(11): 2006-2018, 2009.

[6] B. Xin, J. Chen, J. Zhang, L. H. Dou, and Z. H. Peng, Efficient decision-makings for dynamic weapon-target assignment by virtual permutation and tabu search heuristics, *IEEE Trans. Syst. Man Cybern. C, Appl. Rev.*, 40(6): 649–662, 2010.

[7] B. Xin, J. Chen, Z. H. Peng, L. H. Dou, and J. Zhang, An efficient rule-based constructive heuristic to solve dynamic weapon-target assignment problem, *IEEE Trans. Syst. Man Cybern. A, Syst. Human.*, 41(3): 598–606, 2011.

[8] P. A. Hosein and M. Athans, Some analytical results for the dynamic weapon-target allocation problem, MIT Laboratory for Information and Decision Systems with partial support, USA, Tech. Rep. LIDS-P-1944, 1990.

[9] D. Khosla, Hybrid genetic approach for the dynamic weapon-target allocation problem, in *Proc. SPIE*, 4396, Orlando, USA, 2001: 244−259.

[10] B. J. Zacherl, Weapon-target pairing: revising an air tasking order in real-time, Master Thesis, Naval postgraduate School, Monterey, California, 2006.

[11] L. Wu, H. Y. Xing, F. X. Lu, and P. F. Jia, An anytime algorithm based on modified GA for dynamic weapon-target allocation problem, in Proc. IEEE Cong. Evol. Comput., Hong Kong, China, 2008: 2020–2025.

[12] D. Dionne, E. Pogossian, A. Grigoryan, J. Couture, and E. Shahbazian, An optimal sequential optimization approach in application to dynamic weapon allocation in naval warfare, in Proc. 11th Inter. Conf. Info. Fusion, Colgone, Germany, 2008: 1–6.

[13] J. Li, R. Cong, and J. Xiong, Dynamic WTA optimization model of air defense operation of warships' formation, J. Syst. Eng. Elec., 7(1): 126-131, 2006.

[14] P. Larrañaga and J. A. Lozano, Estimation of distribution algorithms: a new tool for evolutionary computation. Boston: Kluwer Academic Publishers, 2002.

[15] M. Pelikan, D. E. Goldberg, and F. G. Lobo, A survey of optimization by building and using probabilistic models, Comput. Optim. Appl., 21(1): 5-20, 2002.

[16] H. Mühlenbein, The equation for response to selection and its use for prediction, Evol. Comput., 5(3):.303–346, 1998.

[17] C. A. C. Coello, Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art, Computer Methods in Applied Mechanics and Engineering, 191(11-12): 1245-1287, 2002.

[18] T. P. Runarsson and X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Trans. Evol. Comput., 4(3): 284-294, 2000.

[19] K. Deb, An efficient constraint handling method for genetic algorithms, Comput. Method. Appl. Mech. Eng., 186(2-4): 311-338, 2000.