2024 European Control Conference (ECC)
June 25-28, 2024. Stockholm, Sweden

Distributed Neighborhood Search Algorithm for Target Assignment

Tianyu Jin, Shaoming He* and Hyo-Sang Shin

Abstract— **This paper investigates the weapon-target assignment problem and a distributed neighborhood search algorithm is proposed to solve this problem. The proposed algorithm is developed based on the very large-scale neighborhood search (VLSN) algorithm, which is originally developed for centralized allocation among agents. We improve the construction of the improvement graph and the search process for valid cycles in the VLSN algorithm. This enables that the allocation algorithm can be deployed in a distributed way. Each missile maintains a local improvement graph by communicating with its neighbors and attempts to search for valid cycles. The valid cycle directs missiles to exchange their attack targets to achieve the distributed target assignment. Extensive numerical simulations demonstrate the effectiveness of the method.**

I. INTRODUCTION

Modern warfare is evolving towards swarm combat, and the most important aspect of it is the reasonable allocation of vehicle resources. The solution of the target allocation problem is a prerequisite for the realization of many-to-many guidance. Some works consider the above two problems together, which constitute the so-called two-stage target assignment and trajectory optimization problem [1], [2]. This paper focuses on the static weapon target assignment (WTA) problem, which is a classical problem in operations research [3]. The WTA problem aims to find the association mapping between missiles and enemy targets, generally by optimizing an objective function that considers the current situation. Through properly designing the objective function, we can consider different criteria, e.g., energy consumption, advantageous homing geometry, interception time, in the association task [2], [4], [5].

It is known that the WTA problem is an NP-complete problem [6], which indicates that it is impossible to find an optimal solution with polynomial time complexity. Therefore, obtaining high quality solutions efficiently is one of the most important research directions of WTA. Existing algorithms can be classified as centralized and distributed solutions. Centralized algorithms have a simple architecture and are generally faster in computation, but cannot handle a single point of failure. Distributed algorithms are suitable to be deployed in swarm systems, but require the design of the communication process.

This work was supported by National Natural Science Foundation of China under Grant No. 52302449.

Tianyu Jin and Shaoming He are with the School of Aerospace Engineering, Beijing Institute of Technology, Beijing 100081, China.

Shaoming He is also with Yangtze River Delta Graduate School of Beijing Institute of Technology, Jiaxing 314019, China.

Hyo-Sang Shin is with the School of Aerospace, Transport and Manufacturing, Cranfield University, Cranfield MK430AL, UK.

Corresponding Author. Email: shaoming.he@bit.edu.cn

There are three main types of centralized algorithms, including exact, heuristic and meta-heuristic algorithms. Exact algorithms, such as branch-and-bound and column generation, can obtain optimal solutions, but are computationally intensive and difficult to handle large-scale problems. The exponentially growing solution space makes the computation tricky unless the problem is simplified using some assumptions. For example, assuming that the missiles are homogeneous [7]. Heuristic algorithms usually design some specific computational rules based on the problem structure to solve it. Representative approaches are, for example, marginal-return-based method [8] or submodular optimization [9], and very large-scale neighborhood search (VLSN) [10]. These algorithms can obtain satisfactory solutions in a relatively short time, making them prevalent. Meta-heuristic algorithms are a general class of methods, such as Genetic Algorithms (GA) and Particle Swarm Optimization (PSO), etc. Their key feature is that there are no requirements on the form of the objective function and they can solve medium-sized problems effectively.

The realistic requirements of swarm combat drive the transition from centralized to distributed WTA algorithms. However, due to the complexity caused by the nonlinearity of the WTA problem, the distributed algorithm cannot be obtained by simple modifications of the centralized algorithms. The authors in [11] proposes a task swap-based approach, which is based on duality theory and Dijkstra's algorithm. Based on this, [12] suggested to improve the task swap-based approach by removing the restriction of weights in the graph, thus speeding up the optimization process. The above algorithms can be viewed as distributed variants of heuristic algorithms. In contrast, [13] developed a gradient-based primal-dual optimization method. Specifically, each agent only optimizes its own primal and dual variables, while the other values are obtained through communication. The advantages of this algorithm are that it is more robust and allows for data lag with asynchronous communication since it is a continuous optimization. However, its computational speed is slower than task swap-based methods. In summary, these methods either have average performance or are computationally burdensome. Therefore, we attempt to develop a distributed algorithm with low computational cost and satisfactory performance.

In this paper, we investigate a distributed weapon target assignment method based on VLSN algorithm, called distributed neighborhood search (DNS). The construction of the improvement graph and the search process for valid cycles in VLSN are improved to a distributed implementation. Briefly, we let each missile search for valid cycles by maintaining

a local improvement graph and communicating with its neighbors, and then exchanging targets with appropriate other missiles to achieve a distributed target assignment process. Extensive numerical simulations demonstrate the effectiveness of the proposed method.

The remainder of this paper is organized as follows: Sec. II introduces some preliminary mathematical models and the original VLSN method. Sec. III presents the details of the DNS algorithm. Finally, some simulation results and discussions are shown in Sec. IV.

II. PRELIMINARIES AND BACKGROUNDS

A. Mathematical model of missile-target assignment

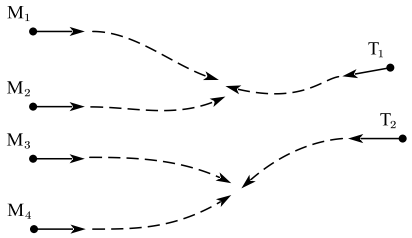

Fig. 1: Multi-missile and multi-target engagement scenario.

The multi-missile-target engagement scenario considered in this paper is shown in Fig. 1. Assume that there are M missiles and N targets involved in the engagement. They are indexed with M_i $(i = 1, 2, ..., M)$ and T_j $(j = 1, 2, ..., N)$. For notational simplicity, we use j as the target index in the remainder of this paper.

In the engagement, every missile intercepts the target with a certain probability. Generally, different missiles have different interception probabilities for a given target, depending on different relative geometries. Multiple missiles cooperation provides the capability to increase the interception probability. This can be mathematically described as

$$p_j = \prod_{i=1}^{M} (1 - p_{ij})^{x_{ij}} = \prod_{i=1}^{M} q_{ij}^{x_{ij}} \tag{1}$$

where p_j denotes the survival probability of target j, p_{ij} represents the conditional probability that the ith missile destroys the jth target when j survives all other missiles. Computing p_{ij} in practice is difficult because it involves many factors. The physical meaning of p_{ij} and the effective calculation methods can be found in [2], [14], [15]. q_{ij} denotes the survival probability of the jth target pursued by missile i. $x_{ij} \in \{0, 1\}$ is a binary variable with $x_{ij} = 1$ indicating that missile i is assigned to target j, and vice versa.

In realistic engagements, targets often consist of various types of vehicles, even including decoys. It is unreasonable to allocate a large number of missiles to low performance vehicles or decoys. Therefore, targets are uniformly described by *values*. Combining target values and interception probabilities, we can calculate the value-weighted survival effectiveness V_j' of the target j:

$$V_j' = V_j p_j \tag{2}$$

where V_j is the value of target j.

Now we can formulate the WTA problem as Problem 1:

$$\begin{aligned} \min_{x_{ij}} J &= \sum_{j=1}^{N} V_j' = \sum_{j=1}^{N} V_j \prod_{i=1}^{M} (1 - p_{ij})^{x_{ij}} \\ \text{s.t.} \quad & \sum_{j=1}^{N} x_{ij} = 1, \quad \forall i = 1, ..., M \\ & x_{ij} \in \{0, 1\}, \quad \forall i = 1, ..., M, \quad \forall j = 1, ..., N \end{aligned} \tag{3}$$

where the equality constraint $\sum_{j=1}^{N} x_{ij} = 1$ ensures that one missile can only be assigned to one target. Note that Eq. 3 is the classical form of the objective function in the WTA problem, which was originally proposed by [3].

B. The Very Large-scale Neighborhood Search Algorithm

To solve large-scale task allocation problems, heuristic algorithms are generally prioritized. The very Large-scale neighborhood search algorithm (VLSN) is one of the most effective algorithms for solving the WTA problem [10]. We first briefly describe VLSN algorithm and then introduce its distributed variant in the next section.

We can consider each missile as a *node* i in the graph, and each target as a group j. When the missile i is assigned to target j, the node i is in group j, denoted as $j = g(i)$. Therefore, the WTA problem is transformed into a partition problem. The feasible solution can be expressed as $S = \{S_1, S_2, \ldots, S_N\}$. $S_j \in S$ is the set of missiles assigned to target j. The calculation of the cost $J(S)$ is the same as the predefined objective function J and thus satisfies $J(S) = \sum_{j=1}^{N} J(S_j)$. To optimize $J(S)$, the VLSN algorithm starts with an initial feasible solution S_0 and continuously improves it until a locally optimal solution is found. To this end, the *improved graph* is introduced, which is a graph where each node i is connected to the other nodes k by arcs, with $g(i) \neq g(k)$. The meaning of arc (i, k) is the action that missile i is reassigned to target $g(k)$ and missile k is removed from $g(k)$. So the cost of this arc (say c_{ik}) is equal to the change of the objective function value due to the above action, which can be expressed as

$$c_{ik} = J\left(\{i\} \cup S_{g(k)} \backslash \{k\}\right) - J\left(S_{g(k)}\right) \tag{4}$$

After constructing the improvement graph, we can find many directed cycles $i_1 - i_2 - \cdots - i_r - i_1$ in it, called subset disjoint cycles. A subset disjoint cycle represents a process of missile reassignment (called cyclic multi-exchange), where no missiles are omitted, as shown in Fig. 2. The negative subset disjoint cycle is called the valid cycle. It can be found that executing a valid cycle multi-exchange can lead to an improvement in the objective function J. The authors in [16] proposes a dynamic programming based approach to find the valid cycle. It first searches for all the negative arcs in the improvement graph and then checks whether they can be closed into valid cycles independently. If so, it performs cycle multi-exchange respectively, called 2-exchange because there are 2 nodes involved. Otherwise, it searches for subsequent

arcs that can make the cost sum of these negative arcs still negative and adds them, then repeats the above operation until no valid cycle exists.

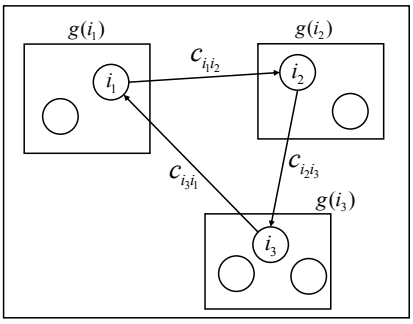

Fig. 2: The subset disjoint cycle exchange.

III. Distributed Neighborhood Search Algorithm

In this section, we develop a distributed neighborhood search algorithm (DNS) based on VLSN. The principle of this algorithm is similar to VLSN, but the implementation is different. The following assumptions need to be made first:

1) The missiles are not too far apart, so the communication topology of missiles is fully connected, meaning that each missile can receive information from all other missiles.
2) The communication process is synchronized.
3) Information of targets are all accessible.

We divide the DNS algorithm into 4 main phases to describe the computational process more clearly:

1) Situational awareness and initial assignment generation.
2) Calculate the costs of arcs.
3) Finding the negative cycles.
4) Reassignment and information updates.

A. Phase 1: Situational awareness and initial assignment generation

In Phase 1, the missile swarm first conducts situational awareness of all targets to obtain information such as target value V_j, interception probability p_{ij}, etc. Then, each missile i computes $q_i = [q_{i1}, q_{i2}, \ldots, q_{iN}]$ for subsequent information exchange.

Recall that the VLSN algorithm optimizes the objective function by improving the initial feasible solution. Therefore, the DNS algorithm also requires an initial assignment for subsequent computations. In this paper, we use a distributed greedy (DG) algorithm [17] to generate the initial allocation. The DG algorithm is based on the submodular optimization theory, in which each missile greedily selects a target, so that a feasible solution can be generated very quickly.

We define all nodes that are not in the same group as node i to be its neighbors, denoted by $\mathcal{N}_i$, and nodes that are within the same group by $\mathcal{N}_i{}'$. The elements in $\mathcal{N}_i$ and $\mathcal{N}_i{}'$ are the indexes of these nodes, respectively.

B. Phase 2: Calculate the costs of arcs

After each missile selects a target, the next step is to calculate the arc costs in the improvement graph. For a node i, an arc either points to i or its neighbors k. Thus only local improvement graphs can be computed for each node, which contain the nodes connected to i.

In phase 2, each node i first exchanges information containing $q_{r,g(i)}$ with nodes $r \in \mathcal{N}_i{}'$. From this each node can calculate $V'_{g(i)}$. Subsequently, i communicates with its neighbor $k \in \mathcal{N}_i$ and receives the index k, the interception data $q_{k,g(k)}$, $q_{k,g(i)}$ and $V'_{g(k)}$. Now each node can compute costs of the arcs with Eq.(5) and construct the local improvement graph.

$$\begin{aligned} c_{ki} &= V'_{g(i)} \left[\frac{q_{k,g(i)}}{q_{i,g(i)}} - 1 \right] \\ c_{ik} &= V'_{g(k)} \left[\frac{q_{i,g(k)}}{q_{k,g(k)}} - 1 \right], \quad k \in \mathcal{N}_i \end{aligned} \tag{5}$$

We use dashed arrows to denote the information flow of communication and solid arrows to denote the arcs in the improvement graph. The illustrations of phase 2 are shown in Fig. 3 and 4.

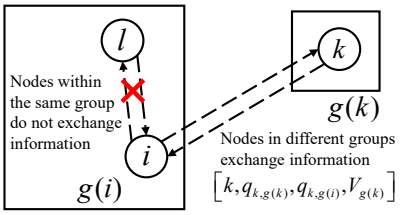

Fig. 3: The flow of information in phase 2.

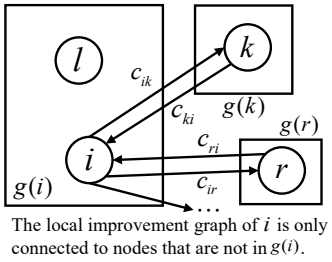

Fig. 4: The local improvement graph of node i.

C. Phase 3: Finding the negative cycles

In phase 3, each node needs to find negative cycles in its own local improvement graph, followed by communicating with neighbors to identify the node that implements 2-exchange.

For node i, its first step is to find all the negative cycles and the corresponding neighborhood node indexes in the local improvement graph:

$$\mathcal{C}_i = \{(c, k) \mid c = c_{ik} + c_{ki} < 0, k \in \mathcal{N}_i\} \tag{6}$$

and find the smallest negative cycle and its corresponding neighbor indexes

$$\mathcal{C}_i^{min} = \min c, \quad c \in \mathcal{C}_i \tag{7}$$

$$\mathcal{I}_i^{min} = \arg\min_k c, \quad c \in \mathcal{C}_i \tag{8}$$

If there is no negative cycle, the above variables are the empty set $\emptyset$. Next, each node exchanges $\mathcal{I}_i^{min}$ with its neighbors. For node i, if there is an entry in its received information that satisfies the following equation

$$\begin{aligned} \mathcal{I}_{k^*}^{min} &= i \\ \mathcal{I}_i^{min} &= k^* \end{aligned} \tag{9}$$

then nodes i and k^* form a profitable 2-exchange (i, k^*), which adds to the set $\mathcal{E}$.

Note that the following special case may occur at this point: consider a 4-node scenario as shown in Fig. 5. Where node 1 and node 2 have been determined to be exchanged, while node 3 has $\mathcal{I}_i^{min} = 1$ and node 4 has $\mathcal{I}_i^{min} = 3$. Since node 1 is already occupied, node 3 can form another pair with node 4 with less improvement. However, node 3 and node 4 each require an additional communication round to obtain this information (node 3 knows from 1 that it has been rejected, while node 4 learns from 3 that it has been selected.), and more communication rounds lead to more time expenses, which should be avoided in distributed systems. Furthermore, extensive experiments have found that negative cycles of length 2 in the improvement graph are sparse, so the special cases rarely arise. In this paper, when the first batch of exchanged node pairs are determined, all the remaining nodes enter phase 4 to complete the subsequent computation.

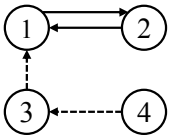

Fig. 5: The illustration of a special case.

D. Phase 4: Reassignment and information updates

The nodes $i \in \mathcal{E}$ exchange their targets based on $\mathcal{E}$, update $g(i)$ and $V_{g(i)}$. Other nodes k (including i) satisfying $g(k) = g(i)$, $i \in \mathcal{E}$ need to update $V'_{g(k)}$. This process is similar to phase 2, except that a significant number of nodes do not need to update their local information. Phase 2 to phase 4 are repeated until there is no profitable 2-exchange, and the algorithm terminates.

IV. SIMULATION RESULTS

In this section, we test the performance of the proposed DNS algorithm in detail and compare it with the VLSN and DG algorithm. Since it is difficult to obtain optimal solutions for large-scale WTA problems, we use the dataset provided

Algorithm 1 The distributed neighborhood search algorithm.

Input: The numbers of missiles and targets, M, N, the interception probabilities p_{ij} $(i = 1, ..., M,\ j = 1, ..., N)$.
Output: The local optimal solution of the problem.

1: Each missile node i acquires V_j, p_{ij} and computes q_i.
2: Generate initial assignment solution with DG algorithm, each node i obtains the index $g(i) = j$ of the attack target.
3: Each node i communicates with other nodes to initialize the set of neighbors $\mathcal{N}_i$ and $\mathcal{N}_i'$.
4: Each node i communicates with node $r \in \mathcal{N}_i'$ to obtain $q_{r,g(i)}$ and then computes $V'_{g(i)}$.
5: Each node i communicates with node $k \in \mathcal{N}_i$ to obtain $\left[k, q_{k,g(k)}, q_{k,g(i)}, V'_{g(k)}\right]$.
6: **repeat**
7: Each node i calculates c_{ik}, c_{ki} through Eq. 5 to construct the local improvement graph, and then calculates $\mathcal{C}_i$ and $\mathcal{I}_i^{min}$ via Eq. 6 and 8, respectively.
8: Each node i communicates with node $k \in \mathcal{N}_i$ to obtain $\mathcal{I}_k^{min}$.
9: **if** $\exists k^*$ such that nodes i^* and k^* satisfy Equation 9 **then**
10: Nodes i^* and k^* exchange their targets, broadcast to other nodes and update $g(i^*)$, $g(k^*)$ and $V_{g(i^*)}$, $V_{g(k^*)}$.
11: Each node k updates $\mathcal{N}_k$, $\mathcal{N}_k'$. Each node $r \in \mathcal{N}'_{i^*}$ updates $V'_{g(i^*)}$ and $l \in \mathcal{N}'_{k^*}$ updates $V'_{g(k^*)}$.
12: Each $k \in \mathcal{N}_{i^*}, \mathcal{N}_{k^*}$ exchanges [k, $q_{k,g(k)}$, $q_{k,g(i)}$, $V'_{g(k)}$] with other nodes $i \in \mathcal{N}_{i^*}, \mathcal{N}_{k^*}$, respectively.
13: **end if**
14: **until** There is no node that satisfies Eq. 9.

in [18], which uses an exact algorithm called column enumeration to obtain optimal solutions under different cases. We consider 19 WTA scenarios with various sizes in the simulations. There are 10 random cases in each scenario, where V_j and p_{ij} are randomly generated.

One of the most important test aspects is the solution quality. The cases in all scenarios are computed using DG, DNS and VLSN algorithms. For convenience of comparison, the maximum cycle lengths in the VLSN algorithm are set to 2 and 5, resulting in 2-exchange and up to 5-exchange, denoted by 2-VLSN and 5-VLSN, respectively. The initial solution of the VLSN algorithm is the same as that used in the DNS in order to be consistent. The simulation results are shown in Table I. The numbers in the first column of the table indicate the total number of missiles and targets, e.g. 10-5 means 10 missiles and 5 targets. G denotes the relative error between the objective function value optimized by the four algorithms and the optimal solution, computed as:

$$G = \frac{|J - J_{opt}|}{J_{opt}} \times 100\% \tag{10}$$

where J is the objective function value optimized by the algorithm and J_{opt} is the optimal value given by the dataset. The detailed calculation of J_{opt} can be found in [18]. G_{min}, G_{aver} and G_{max} denote the minimum, average, and

TABLE I: Relative errors of the three algorithms in different scenarios.

Scenarios	DG			DNS			2-VLSN			5-VLSN		
	G_{min}	G_{aver}	G_{max}	G_{min}	G_{aver}	G_{max}	G_{min}	G_{aver}	G_{max}	G_{min}	G_{aver}	G_{max}
5-5	0	5.7767	19.5663	0	0.8295	7.3795	0	0.8295	7.3795	0	0	0
10-5	0	10.2554	23.5573	0	1.3233	8.3511	0	2.6791	8.3511	0	0.1732	1.7318
10-10	0	8.2363	17.1219	0	2.5765	7.7141	0	3.3725	10.8271	0	0	0
10-20	0	1.2720	2.6352	0	0.6153	1.9944	0	0.6475	1.9944	0	0.2653	1.3164
20-10	2.4585	12.8273	17.8152	0.8844	4.8648	12.3096	0.8844	4.7626	12.3096	0	0.0221	0.2207
20-20	3.3401	8.4442	15.1178	1.2409	2.9256	5.1231	2.1018	3.8822	7.6587	0	0.2596	1.5735
40-10	7.6179	14.8816	26.1042	0.1696	2.9405	6.0135	0.1696	5.1079	11.2663	0	0.0208	0.1125
40-20	6.7309	13.2484	17.5487	1.7272	5.2323	10.2647	1.7272	5.6791	12.5411	0	0.1329	0.4188
40-40	4.3879	6.0311	7.5358	1.9500	2.9740	4.8951	1.7877	3.0863	4.4033	0	0.2373	0.6170
80-20	10.0870	14.6080	18.2556	3.4197	6.4468	9.4595	4.0600	6.8578	8.8515	0	0.0932	0.4236
80-40	4.4582	7.6826	11.2750	2.6827	3.7652	5.7879	2.1171	4.2323	6.0996	0.0110	0.1927	0.4943
80-80	2.6025	3.6304	4.7267	1.1019	1.8904	2.6690	1.2935	2.0981	3.1289	0.0472	0.1551	0.3567
80-320	0.1330	0.2037	0.2883	0.1119	0.1617	0.2170	0.1119	0.1660	0.2226	0.0321	0.0747	0.1196
100-50	4.4410	6.0092	7.9147	1.9378	3.0216	4.3216	2.7571	3.4879	4.9035	0.0325	0.1837	0.3701
100-100	2.3515	3.3727	4.3613	0.9530	1.6709	2.0415	1.0634	1.7792	2.5397	0.0199	0.2817	1.1277
100-200	0.4523	0.6132	0.8218	0.2831	0.4794	0.6389	0.3573	0.4936	0.6389	0.0975	0.2185	0.4350
200-100	3.3920	4.0683	4.6308	1.8875	2.3644	2.7474	1.8743	2.6288	2.9769	0.0514	0.1912	0.3198
200-200	1.8765	2.1899	2.6516	1.0892	1.2617	1.4412	1.0913	1.3417	1.5120	0.0819	0.1910	0.2953
200-400	0.2843	0.3844	0.4878	0.2572	0.3071	0.3694	0.2654	0.3223	0.3934	0.0798	0.1225	0.2360

maximum relative error among the 10 cases in each scenario, respectively. Percentage signs are omitted from the table for convenience.

It can be found that DNS substantially outperforms the DG algorithm on all scenarios, which is due to the fact that DNS improves the solution obtained from DG algorithm. In addition, the DG algorithm has been proved to have a guaranteed lower bound on its performance [17] so that the DNS algorithm will not be worse than this value if there are no external disturbances such as communication and data transmission problems, indicating that the DNS algorithm is effective in dealing with the WTA problem.

Compared to the 2-VLSN algorithm, DNS outperforms 2-VLSN in the vast majority of scenarios. In each iteration, DNS may execute multiple profitable 2-exchange in parallel, whereas 2-VLSN performs only one 2-exchange at a time. Intuitively, the former is not necessarily better than the latter because each 2-exchange is based on the current improvement graph, and performing a 2-exchange possibly eliminates other profitable 2-exchange. However, the results show that DNS has better performance by executing multiple 2-exchange simultaneously.

5-VLSN is the best performing algorithm because the longer cycle length allows larger neighborhoods to be searched. However, it is difficult to implement n-exchange with a length greater than 2 in a distributed system, so this paper only focuses on 2-exchange.

Define $\kappa = M/N$ to be the ratio of the number of missiles to the number of targets. Note that different κ has a large impact on the performance of the DNS algorithm. The performance of DNS decreases when κ becomes large. This is possibly due to the fact that larger κ leads to complexity in the solution space, and therefore it is more difficult to search for profitable exchanges. Another possible reason is that the initial solution provided by DG algorithm is poor when κ is large, causing the DNS to suffer. A suitable example is scenarios 80-20 and 80-320. Both the DG and DNS algorithms perform poorly in the former and better in the latter. The results imply that κ can measure the complexity of the WTA problem to some extent. When $\kappa \leq 1$, each missile individually chooses a target resulting in the highest payoff, hence the better solution is easy to find. When $\kappa > 1$, the better solution is difficult to find because there exist targets that will be selected by more than one missile, leading to the nonlinear term in Eq. (2).

Next, we evaluate the number of iterations, which reflects the speed of convergence of the algorithms and also directly affects the computation time. Table II shows the iteration number (denoted by T) for the DNS and the original VLSN algorithm in all scenarios. For the DNS algorithm, we count the number of cycles between phases 2 and 4, corresponding to lines 6 to 14 of Algorithm 1. Note that the DNS requires an additional round of communication to determine that there are no profitable 2-exchange in the improvement graph, so its minimum number of iterations in the first few small-scale scenarios is 1. It can be found that the number of iterations is much less than VLSN due to the parallel 2-exchange feature of the DNS algorithm. As the size of the scenario becomes larger, the iteration number of the VLSN increases considerably, leading to longer computation times. In addition, κ affects the performance and efficiency of the algorithm, which is the same as in the previous analysis. Another thing worth noting is that larger scale scenarios have smaller relative errors instead. This is probably due to the fact that the solution space of the large-scale problem has more local optimal solutions which do not have a large gap to the optimal solution.

TABLE II: Number of iterations of the three algorithms in different scenarios.

Scenarios	DNS			2-VLSN			5-VLSN		
	T_{min}	T_{aver}	T_{max}	T_{min}	T_{aver}	T_{max}	T_{min}	T_{aver}	T_{max}
5-5	1	1.7	3	0	0.8	2	0	1.2	3
10-5	1	2	3	0	1.5	5	0	2.1	6
10-10	1	1.9	4	0	1.6	4	0	3.4	8
10-20	1	2.2	4	0	1.5	4	0	2.2	5
20-10	1	3.2	8	0	3.7	8	2	7.6	11
20-20	2	3	5	2	3.9	6	7	9.4	18
40-10	2	3.6	7	2	6.9	11	6	11.7	21
40-20	2	3.9	8	2	8.4	11	9	19.1	26
40-40	2	3.4	5	3	5.9	10	12	18.5	25
80-20	3	4	6	4	11.9	17	23	33.1	46
80-40	2	4.1	6	7	10.5	16	25	37.5	51
80-80	2	3.5	5	3	8.5	12	27	34.8	48
80-320	2	2.6	4	1	4.1	9	15	23.8	33
100-50	2	3.6	5	4	9.6	16	38	45.7	55
100-100	3	4.3	6	9	15.1	23	36	50.4	69
100-200	2	3.2	6	4	8.5	13	24	40.4	54
200-100	3	4.3	5	11	18.9	30	78	91.2	110
200-200	2	3.6	5	9	16.4	29	71	90.4	111
200-400	2	3.6	6	4	11	19	35	82.9	111

V. CONCLUSIONS

This paper proposes a distributed neighborhood search algorithm to solve the weapon-target assignment problem. In the DNS algorithm, each missile maintains a local improvement graph by communicating with neighbors and searches for valid cycles in it. The objective function value is improved when missiles follow the instructions of the valid cycles by exchanging their targets, thus solving the WTA problem in a distributed manner. The DNS algorithm does not require a central node and is fully distributed. Extensive simulation results in various scenarios show that the proposed algorithm can generate satisfactory solutions. Its parallel feature also reduces the number of iterations compared to the original VLSN algorithm. Thus it can solve the WTA problem effectively. Its parallel feature also reduces the number of iterations compared to the original VLSN algorithm. Thus it can solve the WTA problem effectively.

REFERENCES

[1] Weinan Wu, Xiaogang Wang, and Naigang Cui. Fast and coupled solution for cooperative mission planning of multiple heterogeneous unmanned aerial vehicles. *Aerospace Science and Technology*, 79:131–144, 2018.

[2] Tianyu Jin, Shaoming He, Hongyan Li, and Xiaobo Zheng. Evaluation model and exact optimization algorithm in missile–target assignment. *Journal of Guidance, Control, and Dynamics*, pages 1–8, 2023.

[3] Alan S Manne. A target-assignment problem. *Operations research*, 6(3):346–351, 1958.

[4] Asım Tokgöz and Serol Bulkan. Weapon target assignment with combinatorial optimization techniques. *International journal of advanced research in artificial intelligence*, 2(7):39–50, 2013.

[5] Alexander Kline, Darryl Ahner, and Raymond Hill. The weapon-target assignment problem. *Computers & Operations Research*, 105:226–236, 2019.

[6] Stuart P Lloyd and Hans S Witsenhausen. Weapons allocation is np-complete. In *1986 summer computer simulation conference*, pages 1054–1058, 1986.

[7] Kyle Volle, Jonathan Rogers, and Kevin Brink. Decentralized cooperative control methods for the modified weapon–target assignment problem. *Journal of Guidance, Control, and Dynamics*, 39(9):1934–1948, 2016.

[8] B. Xin, Y. Wang, and J. Chen. An efficient marginal-return-based constructive heuristic to solve the sensorweapontarget assignment problem. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(12):2536–2547, 2019.

[9] Zengfu Wang, Xuezhi Wang, Yan Liang, and Quan Pan. Weapon target assignment leveraging strong submodularity. In *2013 IEEE International Conference on Information and Automation (ICIA)*, pages 74–79. IEEE, 2013.

[10] Ravindra K Ahuja, Arvind Kumar, Krishna C Jha, and James B Orlin. Exact and heuristic algorithms for the weapon-target assignment problem. *Operations research*, 55(6):1136–1146, 2007.

[11] Lantao Liu and Dylan A Shell. A distributable and computation-flexible assignment algorithm: From local task swapping to global optimality. In *Proc. Robot.: Sci. Syst.*, pages 257–264, 2013.

[12] Peng Zhao, Jianzhong Wang, Lingren Kong, et al. Decentralized algorithms for weapon-target assignment in swarming combat system. *Mathematical Problems in Engineering*, 2019, 2019.

[13] Katherine Hendrickson, Prashant Ganesh, Kyle Volle, Paul Buzaud, Kevin Brink, and Matthew Hale. Decentralized weapon–target assignment under asynchronous communications. *Journal of Guidance, Control, and Dynamics*, 46(2):312–324, 2023.

[14] Weilin Luo, Jinhu L, Kexin Liu, and Lei Chen. Learning-based policy optimization for adversarial missile-target assignment. *IEEE Transactions on Systems Man Cybernetics-Systems*, 52(7):4426–4437, 2022.

[15] Zongyuan Sun and Jianying Yang. Multi-missile interception for multi-targets: Dynamic situation assessment, target allocation and cooperative interception in groups. *Journal of the Franklin Institute*, 359(12):5991–6022, 2022.

[16] Ravindra K Ahuja, James B Orlin, and Dushyant Sharma. A composite very large-scale neighborhood structure for the capacitated minimum spanning tree problem. *Operations Research Letters*, 31(3):185–194, 2003.

[17] Guannan Qu, Dave Brown, and Na Li. Distributed greedy algorithm for multi-agent task assignment problem with submodular utility functions. *Automatica*, 105:206–215, 2019.

[18] Yiping Lu and Danny Z. Chen. A new exact algorithm for the weapon-target assignment problem. *Omega*, 98:102138, 2021.