Efficient Adaptive Large Neighborhood Search for Sensor–Weapon–Target Assignment

Yang Wang, Junpeng Wang, Jin-Kao Hao, Jianguang Feng

***Abstract*—We study a sensor-weapon-target assignment (S-WTA) problem that considers the desired probability of target destruction and aims to minimize the total cost of combat resources. Lower and upper bounds for the S-WTA problem are obtained by constructing linear approximation models. We also propose an adaptive large neighborhood search (ALNS) algorithm characterized by a model-driven repair phase to solve this problem. The destruction phase adaptively selects a destruction operator to remove partial resource assignments and produces an incomplete reference solution. For the destroyed solution, the repair phase generates a reduced subproblem that optimizes only the destroyed parts while keeping the other parts fixed. Each subproblem is formulated as a mixed integer programming model and solved by a general-purpose solver to repair the destroyed solution. Computational experiments show that the approximation formulations can obtain tight lower and upper bounds for most problem instances. Moreover, our proposed ALNS algorithm is competitive with the solver for small instances and effectively solves large instances. In addition, we experimentally demonstrate that our ALNS outperforms state-of-the-art algorithms in the literature, and the proposed model-driven solution repair phase outperforms the traditional heuristic repair operators.**

***Index Terms*—combinatorial optimization, sensor-weapon-target assignment, nonlinear constraint, adaptive large neighborhood search.**

I. INTRODUCTION

IN the modern battlefield, both attackers and defenders use various offensive and defensive military equipment. The defender is supposed to have a group of weapons to intercept a series of targets from the attacker. Due to the various attributes of weapons and targets, allocating different weapons to a particular target could lead to differing probabilities of destruction and/or varying assignment costs. Over the past decades, constructing an optimal assignment scheme to achieve specific military goals has received considerable attention. This problem is known as the Weapon Target Assignment (WTA) problem [1]. The WTA problem has been proved to be NP-complete [2]. To simplify the problem, denBroeder *et al.* [3] studied a special case where all weapons are assumed to have the same killing probability for the targets. Kwon *et al.* [4] constructed a WTA problem with a nonlinear constraint and employed the logarithmic transformation for linearization.

Network-centric warfare has become an emerging ideology for military construction. To rapidly guide weapons toward threatening targets, commanders need to collect battle information efficiently. As a result, various sensors play a pivotal role through their detection capabilities. Bogdanowicz *et al.* [5] introduced the first Sensor Weapon Target Assignment (S-WTA) problem, which considered the assignment of sensors in addition to the weapons. Due to the high complexity, heuristic algorithms have been applied for solving various S-WTA problems [6], [7].

Most of the literature on the WTA and S-WTA is concerned with minimizing the expected threat value of destroying targets. In general, the threat of a specific target stems from the damage it can inflict on the objects protected by the defender. Although the minimization of the total expected threat value is a key objective for measuring the quality of operational engagement, evaluating the threat value of all targets proves challenging in numerous defense scenarios. Specifically, comparing the value of diverse objects under defender protection, such as populations, facilities, or assets, can be complicated. Hence, it's more practical for commanders to use a minimum desired destruction probability for each target as an additional guideline. Meanwhile, as combat resources become more expensive, good command controlling the operational cost plays a key role in weaponry. In addition, the truly complex battlefield environments require high time response and numerous combat resources and threatening targets significantly expand the decision-making framework. Hence, we investigate a novel S-WTA problem that accommodates these conditions and develop efficient algorithms to obtain high-quality solutions.

The S-WTA problem studied in this paper aims to minimize the total cost of assigned combat resources so that each target's minimum desired destruction probability is respected. Specifically, the proposed S-WTA model has a linear objective that minimizes the sum of the assignment cost of both sensors and weapons. The requirement to achieve a specific level of destruction for all targets imposes nonlinear constraints, significantly improving the complexity of solving this problem. We note that some studies [4], [8] also considered these features in their work; however, they did not consider the joint allocation of weapons and sensors. In addition to the introduced S-WTA problem, this work has the following main contributions.

This work was supported in part by the National Natural Science Foundation of China, under Grants 71971172, 71901176 and 72371200, and the Open Research Subject of State Key Laboratory of Intelligent Game, under Grant ZBKF-24-16 and ZBKF-23-02. (Corresponding author: Jianguang Feng.)

Yang Wang, Junpeng Wang, and Jianguang Feng are with the School of Management, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: yangw@nwpu.edu.cn; wangjunpeng97@mail.nwpu.edu.cn; feng@nwpu.edu.cn). Jin-Kao Hao is with the Department of Computer Science, LERIA Lab, Université d'Angers, Angers 49045, France (e-mail: jin-kao.hao@univ-angers.fr).

- We present an integer nonlinear formulation of the S-WTA. Given its nonlinear nature, we provide a comprehensive analysis of the problem and derive its lower and upper bounds by exploring the linear approximations of the nonlinear formulation. Experimental results showed that the approximation formulations could obtain relatively tight bounds for most of the instances.
- To handle large and challenging instances, we propose an adaptive large neighborhood search (ALNS) algorithm that alternatively executes an adaptive destruction phase and a model-driven repair phase. The destruction phase employs six destruction operators specifically designed for handling the assignment of weapons, sensors, or both. For the destroyed solution, the repair phase constructs a reduced subproblem that only optimizes the destroyed assignments while keeping the others fixed.
- We conduct comprehensive numerical experiments to evaluate the performance of the proposed ALNS algorithm. This is achieved by comparing it with four reference methods, including the (UB) model, a variant of the ALNS algorithm that employs traditional heuristic repair operators, the MRBCH algorithm [6], and the VLSN algorithm [9]. Computational results indicate that ALNS is competitive with the (UB) model for small instances and outperforms it for large instances. Meanwhile, ALNS is able to obtain much better objective values than the other reference algorithms.

The remainder of this paper is organized as follows. Section II presents a literature review of WTA and S-WTA problems. Section III formally describes the studied S-WTA problem and formulates it as an integer nonlinear model. In Section IV, we derive the lower and upper bounds of the S-WTA problem by exploring linear approximations of the nonlinear model. Section V presents an effective adaptive large neighborhood search algorithm specifically developed for the S-WTA problem. In Section VI, computational experiments are conducted to evaluate the performance of the proposed linear approximations and ALNS algorithm. Finally, Section VII concludes the work.

II. Related works

This section constructs a brief literature review of WTA and S-WTA problems. Most studies classify the problems into static and dynamic problems, where the dynamic version considers uncertain information that emerges over time, i.e., new targets or available resources. In this paper, we mainly focus on static problems.

Since the first WTA problem was introduced [1], many WTA variants have been studied. Several exact algorithms are explored to obtain the optimal problem solution efficiently. Kwon *et al*. [4], [8] studied the WTA problem with a linear objective function to minimize the total firing cost, provided that each target's minimum desired probability level is respected. The authors developed a lagrangian relaxation approach and a follow-up branch-and-price approach to obtain exact solutions to this problem. Ahuja *et al*. [9] adopted a piecewise linear approximation of the nonlinear objective function, i.e., the minimization of the total expected threat value of all targets, to determine the lower bounds of the problem. They also developed a branch-and-bound algorithm that utilizes three lower-bounding schemes to tackle this problem. In recent years, Kline *et al*. [10] proposed a nonlinear branch-and-bound algorithm for solving the WTA problem to minimize the same objective as in [9]. An NLP solver is used to solve a variant of the integer-relaxed formulation to obtain bounds. Lu *et al*. [11] constructed a linear formulation of the WTA problem to minimize the threat value of targets. They innovatively represent each integer variable by a sequence of binary bits, where each bit determines whether the weapon is assigned to a target or not. They proposed a column enumeration algorithm, which significantly reduces the number of columns by limiting the number of weapons assigned to the targets and exploring the dominant relationships between them. Andersen *et al*. [12] studied a traditional WTA problem to minimize the total threat value. Based on several existing linear approximation approaches, they used convex piecewise linear functions to linearize the objective function and proposed a branch-and-adjust algorithm for solving the WTA problem.

To solve large scale WTA problems within a short time, most studies resort to high-performance heuristics. Shang *et al*. [13] investigated a WTA problem that minimizes the total loss of protected assets. They proposed an immune genetic algorithm where local information is utilized in the crossover and mutation operations. Ahuja *et al*. [9] proposed a very large-scale neighborhood search algorithm to obtain near-optimal solutions. The search initiates with a feasible solution to the WTA problem and proceeds through a series of "cyclic and path exchanges" to improve the solution. Sonuc et al. [14] explored the same WTA problem as in [9] except that an equal number of weapons and targets are required. They proposed a parallel version of the simulated algorithm in which a multi-start technique is employed to restart a heuristic algorithm with different configurations to improve the solution quality. Kline *et al*. [10] developed a modified quiz problem search algorithm to tackle large instances of the WTA problem. Subsequently, the authors devised an Eminent Domain metaheuristic based on the quiz problem heuristic to solve the same problem [15]. This method begins with a given initial feasible solution and iteratively denies assignment pairings within a subset of weapon-target pairs, maximizing the overall benefit by reallocating these assignments to other potential pairings. Some other methods include learning algorithms [16], [17], swarm intelligence algorithms [18]–[21], quantum algorithms [22], game theory [23], and hybrid heuristics [24], [25]. For a comprehensive review of various WTA problems, the interested reader is referred to Kline *et al*. [26].

In the network-centric warfare context, substantial studies have also considered allocating sensors in the WTA problem. Some representative exact algorithms for solving S-WTA problems are presented as follows. Bogdanowicz and Coleman [5] studied a simplified S-WTA problem, whose linear objective is to maximize the total benefit of assigning each sensor to each target and each weapon to each target. They proposed a forward auction with the benefit scaling algorithm to solve this S-WTA problem. Bogdanowicz [27] further studied a symmetric assignment problem and considered the same numbers of sensors, weapons, and targets. Based on above auction algorithm, the proposed Swt-opt algorithm converges to an optimal solution in a finite number of steps. Given the significant reliance of Swt-opt on network topologies, Li *et al*. [28] improved the Swt-opt algorithm of [27] by additionally introducing a consensus algorithm.

Due to the high computational complexity of the S-WTA problem and its variant, various heuristic and metaheuristic algorithms have been proposed to handle these problems. In the study of Chen *et al*. [29], a particle swarm optimization algorithm is put forward to solve the S-WTA problem. Unlike previous literature on S-WTA, this work features a nonlinear objective aimed at minimizing the total survival probabilities of all targets, while allowing multiple weapons to be assigned to the same target at once. Xin *et al*. [6] constructed an S-WTA model that considers the minimization of the total threat value for all targets. They developed a marginal-return-based constructive heuristic, where each iteration selects one sensor–weapon–target triplet and exploits problem-specific knowledge contained in the marginal returns. Li *et al*. [7] studied a synthetical Sensor-Weapon-Target model with the objective of maximizing the expected value of detected threat from sensors and eliminated threat from weapons. They developed a novel genetic algorithm to address this problem where a problem-specific population initialization method based on prior knowledge is designed. Zhang *et al*. [30] studied an S-WTA problem in ground-to-air defense, which considers the interdependencies of sensors and heterogeneous weapons. The minimization of total expected threat value from targets is considered in their model. They proposed a novel evolutionary algorithm to obtain the sequential interception schemes and adopted three special coding methods to build the population individuals. In summary, the previous works show that: a) no study has addressed the minimum desired probability of target destruction when conducting the joint allocation of weapons and sensors. The requirement to achieve a specific level of destruction for all targets imposes nonlinear constraints, significantly improving the complexity of solving this problem, and b) many heuristic algorithms are well designed in the literature, yet there is a lack of methods to establish the bounds for the sensor-weapon-target assignment problem. In this paper, we fill the gaps by formulating a new problem in network-centric warfare and constructing linear approximation models to obtain valid upper and lower bounds.

III. Problem description and formulation

We are given a set of sensors S, a set of weapons W and a set of targets T. Let p_{ik} denote the probability of target $k \in T$ being detected by sensor $i \in S$, and let q_{jk} represent the probability of target $k \in T$ being destroyed by weapon j, provided that the target has been detected. Let a_{ik} and b_{jk} denote the cost of assigning sensor $i \in S$ and weapon $j \in W$ to target $k \in T$, respectively. A minimum destruction probability u is desired for each target. The aim is to assign weapons and sensors to targets to minimize the total assignment cost while ensuring minimum desired destruction probability for each target.

To formulate this problem, we define two sets of binary decision variables, x_{ik} and y_{jk}, where x_{ik} takes 1 if sensor i is assigned to target k and 0 otherwise, and y_{jk} takes 1 if weapon j is assigned to target k and 0 otherwise. Thus, the probabilities of target k being detected and destroyed can be expressed as $A_k = 1 - \prod_{i\in S}(1-p_{ik})^{x_{ik}}$ and $B_k = 1 - \prod_{j\in W}(1-q_{jk})^{y_{jk}}$, respectively. Therefore, we can formulate the S-WTA problem as the following integer nonlinear formulation.

$$\text{(S-WTA)}\ \min \sum_{i\in S}\sum_{k\in T} a_{ik}x_{ik} + \sum_{j\in W}\sum_{k\in T} b_{jk}y_{jk} \quad (1)$$

$$\text{s.t.} \sum_{k\in T} x_{ik} \le 1, \forall i \in S. \quad (2)$$

$$\sum_{k\in T} y_{jk} \le 1, \forall j \in W. \quad (3)$$

$$\sum_{i\in S} x_{ik} \ge 1, \forall k \in T. \quad (4)$$

$$\sum_{j\in W} y_{jk} \ge 1, \forall k \in T. \quad (5)$$

$$A_k B_k \ge u, \forall k \in T. \quad (6)$$

$$x_{ik} \in \{0,1\}, \forall i \in S, k \in T. \quad (7)$$

$$y_{jk} \in \{0,1\}, \forall j \in W, k \in T. \quad (8)$$

The objective (1) is to minimize the total cost of assigning sensors and weapons to targets. Constraints (2) and (3) ensure that each sensor or weapon can only be assigned to at most one target, respectively. Constraints (4) and (5) ensure that at least one sensor and one weapon are assigned to each target, respectively. Constraint (6) guarantees that the desired destruction probability of each target is satisfied. Finally, constraints (7) and (8) define the domains of decision variables.

IV. Lower and upper bounds

In this section, we will first comprehensively analyze the S-WTA problem and then derive its valid lower and upper bounds. We will utilize the original formulation's approximations to achieve this. These linear approximations will not only serve as benchmarks for accessing our heuristic, which is explained in Section VI but will also enhance our heuristic further.

A. Lower bound

We first construct an approximation that yields a valid lower bound. The central concept involves approximating the nonlinear constraint (6) with linear constraints. For this purpose, we restate constraint (6) as follows:

$$A_k \ge u_1 \text{ and } B_k \ge u_2.$$

where (u_1, u_2) belongs to the set $U = \{(u_1,u_2) \mid u_1u_2 \ge u, u \le u_1, u_2 \le 1\}$. Although the set U contains infinite elements, we can approximate constraint (6) using a subset of U. Let $U' \subset U$ be a finite set consisting of $n+1$ elements. To construct set U', we sample a series of values for u_1, which are collected in the set $\{a_0, a_1, \ldots, a_n\}$ and let the corresponding values of u_2 be collected in the set

$\{b_0 = u/a_0, b_1 = u/a_1, \ldots, b_n = u/a_n\}$. A simple way to generate these values is to set $a_l = u + l(1-u)/n$ for each $l \in \{0, 1, \ldots, n\}$. For each target k, we employ a binary variable z_{kl} to indicate which pair of values (u_1, u_2) is active for the target in the solution. Then, we can approximate constraint (6) using the following relationships:

$$A_k \geq \sum_{l=1}^{n} a_{l-1} z_{kl}, \forall k \in T. \tag{9}$$

$$A_k \leq \sum_{l=1}^{n} a_l z_{kl}, \forall k \in T. \tag{10}$$

$$B_k \geq \sum_{l=1}^{n} b_l z_{kl}, \forall k \in T. \tag{11}$$

$$\sum_{l=1}^{n} z_{kl} = 1, \forall k \in T. \tag{12}$$

$$z_{kl} \in \{0, 1\}, \forall l \in \{1, 2, \ldots, n\}, k \in T. \tag{13}$$

Constraints (9) and (10) restrict the probability of target k being detected by sensors, and constraint (11) limits the probability of target k being destroyed by weapons. Constraint (12) requires that exactly one pair of values of (u_1, u_2) is chosen per target. Therefore, replacing constraint (6) of (S-WTA) with constraints (9)–(13) yields:

$$\text{(LB)} \quad \min \quad \sum_{i \in S} \sum_{k \in T} a_{ik} x_{ik} + \sum_{j \in W} \sum_{k \in T} b_{jk} y_{jk}$$
$$\text{s.t.} \quad \text{(2)–(5), (7)–(8), (9)}-\text{(13).}$$

Proposition 1. *The optimal objective value of formulation (LB) is a valid lower bound of the S-WTA problem.*

Proof. See Section I-A of the online supplement. □

Constraints (9)–(11) can be linearized with the following steps. Let's take constraint (9) as an example. Because z_{kl} is a binary variable and constraint (12) restricts only one $z_{kl} = 1$, the following equivalence relations are valid.

Suppose any $z_{kl} = 1$, for this l we get

$$1 - \prod_{i \in S} (1 - p_{ik})^{x_{ik}} \geq a_{l-1}. \tag{14}$$

The above constraint can be linearized using the logarithmic transformation [4], [8]. In particular, by taking the logarithm of both sides of the constraint, we have the following linear constraint:

$$\sum_{i \in S} \ln(1 - p_{ik}) x_{ik} \leq \ln(1 - a_{l-1}). \tag{15}$$

Considering all possible z_{kl}, the final linear constraint is

$$\sum_{i \in S} \ln(1 - p_{ik}) x_{ik} \leq \sum_{l=1}^{n} \ln(1 - a_{l-1}) z_{kl}. \tag{16}$$

We can employ the above procedure to linearize the non-linear constraints (10) and (11).

B. Upper Bound

Similarly to how formulation (LB) is constructed, the following equations yield another approximation of constraint (6).

$$A_k \geq \sum_{l=1}^{n-1} a_l z_{kl}, \forall k \in T. \tag{17}$$

$$A_k \leq \sum_{l=1}^{n-1} a_{l+1} z_{kl}, \forall k \in T. \tag{18}$$

$$B_k \geq \sum_{l=1}^{n-1} b_l z_{kl}, \forall k \in T. \tag{19}$$

$$\sum_{l=1}^{n-1} z_{kl} = 1, \forall k \in T. \tag{20}$$

$$z_{kl} \in \{0, 1\}, \forall l \in \{1, 2, \ldots, n-1\}, k \in T. \tag{21}$$

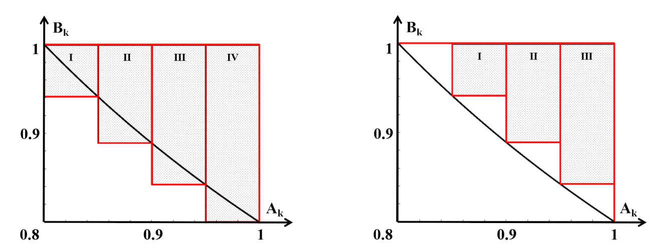
By replacing constraint (6) of model (S-WTA) with (17)–(21), we obtain the following approximation model:

$$\text{(UB)} \quad \min \sum_{i \in S} \sum_{k \in T} a_{ik} x_{ik} + \sum_{j \in W} \sum_{k \in T} b_{jk} y_{jk}$$
$$\text{s.t. Constraints (2)–(5), (7)–(8), (17)–(21).}$$

Proposition 2. *The optimal objective value of formulation (UB) is a valid upper bound of the S-WTA problem.*

Proof. See Section I-B of the online supplement. □

We can utilize the same method of transforming constraint (9) into (16) to linearize the model (UB).



(a) Approximation–LB (b) Approximation–UB

Fig. 1. Linear approximations of the nonlinear constraint ($n = 4$, $u = 0.8$).

Figure 1 illustrates the approximations of constraint (6) with $u = 0.8$. The curve in the figure represents $A_k B_k = u$. The sample set U' is given by $u_1 \in \{0.8, 0.85, 0.9, 0.95, 1\}$ and $u_2 \in \{1, 0.8/0.85, 0.8/0.9, 0.8/0.95, 0.8\}$.The both approximations have $n = 4$ and $n - 1 = 3$ segments, respectively.

V. Solution method

Due to the non-linearity and non-convexity, it is challenging to solve the proposed S-WTA problem exactly. Moreover, heuristics are efficient methods for solving large-scale combinatorial optimization problems, i.e., quadratic assignment problem [31], bandwidth coloring problem [32], service deployment problem [33]. In this section, we present an adaptive large neighborhood search algorithm to find sub-optimal solutions effectively. The proposed algorithm alternates between destruction and repair phases, relying on historical search information and model-driven repair operations. In the destruction phase, an operator is selected from multiple destruction operators and applied to destroy a reference solution. The repair phase optimizes a subproblem that treats the remaining assignments as decision variables and maintains the assignments in the destroyed solution.

The framework of our proposed adaptive large neighborhood search algorithm is outlined in Algorithm 1. At the outset of this algorithm, the associated elements, such as destruction operators and their weights, are initialized (Line 1). Subsequently, an initial feasible solution π^* is constructed using a greedy heuristic. This approach takes into account both the increase in cost and detection/destruction probability of assignments (Line 2). In the destruction phase, one destruction operator θ is selected adaptively from a set of operators D based on the historical performance of each operator (Line 4). This chosen operator is applied to destroy the reference solution π^*, which is the best solution found so far, to generate an incomplete solution π^D (Line 5). More specifically, certain sensors or weapons are removed based on the operators in this phase. Afterward, a reduced-size subproblem is generated, and only its destroyed parts are optimized (Line 6), i.e., the decision variables related to the assignments not involved in π^D while maintaining the other variables at the same values as π^D. Finally, a general solver is used to solve the integer programming model SP, which is constructed for the above subproblem, in order to repair the destroyed solution and obtain a feasible solution π (Line 7). The newly discovered best solutions are recorded and serve as the new reference solutions (Line 9). Meanwhile, the weights of the operators Ω are updated based on their previous performances (Line 11). These above phases are repeated until a predetermined termination criterion, such as reaching a computational time limit or finding the optimal solution (Line 3), is met.

Algorithm 1 Adaptive large neighborhood search scheme

1: Initialize destruction operators D and their weights $\Omega = \{\eta, \dots, \eta\}$, the best found solution $\pi^* \leftarrow \varnothing$ and its objective $f^* \leftarrow \infty$, a given problem instance I.
2: $(\pi^*, f^*) \leftarrow$ `GenerateInitialSolution` (I).
3: **while** the termination criterion is not met **do**
4: $\quad\theta \leftarrow$ `SelectDestructionOperator`(D, Ω).
5: $\quad\pi^D \leftarrow$ `Destroy`(π^*, θ).
6: $\quad$SP $\leftarrow$ `ConstructSubmodel`(π^D).
7: $\quad(\pi, f) \leftarrow$ `Repair`(SP).
8: $\quad$**if** $f < f^*$ **then**
9: $\quad\quad\pi^* \leftarrow \pi$, $f^* \leftarrow f$.
10: $\quad$**end if**
11: $\quad$`Update`(Ω).
12: **end while**
13: **return** (π^*, f^*).

We consider the main procedures in Algorithm 1 to analyze its computational complexity. The initial solution phase adopts a greedy list that includes $O((|S|+|W|)|T|)$ elements. It constructs the initial solution by selecting the elements one by one. Thus its time complexity is bounded by $O((|S| + |W|)|T|)$. The time complexity of the search phase, i.e., line 3 to line 12, is $O(\zeta(\delta(|S| + |W|) + \mu))$, where ζ denotes the total number of iterations, δ is a parameter of destruction level (see Section V-B) and μ denotes the time for submodel solving (see Section V-D). To summarize, the total computational complexity of our algorithm is $O((\zeta\delta + |T|)(|S| + |W|) + \zeta\mu)$.

A. Generation of initial solution

The initial solution is generated by greedily assigning the sensors and weapons to targets until the prescribed destruction probability of each target is satisfied. It mainly consists of the following three steps. First, a sorted list `Glist` $= \{(r,t) \mid r \in S \cup W, t \in T\}$ is constructed with each element (r,t) representing the assignment of resource (a weapon or a sensor) r to target t. The elements in `Glist` are sorted in ascending order of $\alpha_{(r,t)}$ defined by

$$\alpha_{(r,t)} = \begin{cases} \gamma a_{rt} - p_{rt}, & r \in S \\ \gamma b_{rt} - q_{rt}, & r \in W \end{cases} \tag{22}$$

where γ is a given tradeoff parameter. Equation (22) considers the cost and destruction probability of each assignment. Note that a smaller value of γ can help ensure feasibility for instances with a large desired destruction probability u. The first pair (r^*, t^*) in `Glist` is selected according to an *alternative assignment principle* to accommodate the destruction probability requirement. It is clear that the multiplicative form of $A_k \times B_k$ in constraint (6) presents a diminishing marginal utility. Specifically, if A_k is larger than B_k, improving A_k by assigning additional sensors is not as effective as assigning weapons to improve B_k. The alternative assignment principle essentially improves the balance between A_k and B_k. We use an approximation strategy by alternatively assigning weapons and sensors. Let M_k and N_k be the numbers of sensors and weapons assigned to each target, respectively. If $M_k > N_k$, we consider weapons as the next resource to be assigned to the target k, and sensors otherwise, where ties are broken arbitrarily. For each iteration, we take the first pair in `Glist` as (r^*, t^*). If $r^* \in S$ and $M_{t^*} \leq N_{t^*}$, or if $r^* \in W$ and $M_{t^*} \geq N_{t^*}$, (r^*, t^*) satisfies the alternative assignment principle. Otherwise, the next pair in `Glist` is taken as (r^*, t^*) until the principle is satisfied. When no qualified (r^*, t^*) exists and `Glist` is non-empty, we override this rule by simply using the first pair in `Glist`.

Finally, we assign the resource r^* to target t^* and remove the chosen pair (r^*, t^*) from `Glist`. If the target t^* satisfies the minimum destruction probability u, we remove all the pairs (r, t^*) from `Glist`. The selection and assignment of (r^*,t^*) are repeated until all targets satisfy the minimum destruction probability u, resulting in a feasible initial solution.

B. Destruction phase

For the reference solution π^*, the destruction phase employs multiple strategies to destroy a part of this solution, meaning it removes a certain number of sensors and weapons assigned to the targets. Suppose that solution π is represented by the assigned set of sensors π_t^S and the set of assigned weapons π_t^W for each target $t \in T$. To produce a destroyed solution π^D, we propose six destruction operators that work by removing resources or targets, which are referred to as resources destruction operators (R-D, including S-D, W-D and SW-D) or targets destruction operators (T-D, including TS-D, TW-D and TSW-D). Furthermore, the removal of sensors, weapons or both is further distinctively considered. In addition, we probabilistically bias the selection of such solution elements that yield a larger cost reduction when being removed. Specifically, the proposed destruction operators are presented as follows.

- Sensor-Destroy (S-D). This operator removes δ sensors from π^S. Suppose sensor s is assigned to target t, the probability of s being removed, denoted by ψ_s, is calculated by
$$\psi_s = \frac{a_{st}}{\sum_{t \in T} \sum_{s \in \pi_t^S} a_{st}} \tag{23}$$
- Weapon-Destroy (W-D). This operator removes δ weapons from π^W. The probability of weapon w being removed, denoted by ψ_w, is calculated by
$$\psi_w = \frac{b_{wt}}{\sum_{t \in T} \sum_{w \in \pi_t^W} b_{wt}} \tag{24}$$
- Sensor&Weapon-Destroy (SW-D). This operator removes δ sensors and weapons from π^S and π^W, where ψ_s and ψ_w are calculated by
$$\psi_s = \frac{a_{st}}{\sum_{t \in T} \sum_{s \in \pi_t^S} a_{st} + \sum_{t \in T} \sum_{w \in \pi_t^W} b_{wt}} \tag{25}$$
$$\psi_w = \frac{b_{wt}}{\sum_{t \in T} \sum_{s \in \pi_t^S} a_{st} + \sum_{t \in T} \sum_{w \in \pi_t^W} b_{wt}} \tag{26}$$
- Target Sensor-Destroy(TS-D). This operator selects δ targets and removes all sensors assigned to them from π^S. The probability ψ_t^S of target t being selected is calculated by
$$\psi_t^S = \frac{\sum_{s \in \pi_t^S} a_{st}}{\sum_{t \in T} \sum_{s \in \pi_t^S} a_{st}}. \tag{27}$$
- Target Weapon-Destroy(TW-D). This operator selects δ targets and removes all weapons assigned to them from π^W. The probability ψ_t^W of target t being selected is calculated by
$$\psi_t^W = \frac{\sum_{w \in \pi_t^W} b_{wt}}{\sum_{t \in T} \sum_{w \in \pi_t^W} b_{wt}}. \tag{28}$$
- Target Sensor&Weapon-Destroy(TSW-D). This operator selects δ targets and removes all sensors and weapons assigned to them from π^S and π^W. Then the probability ψ_t of target t being selected is
$$\psi_t = \frac{\sum_{s \in \pi_t^S} a_{st} + \sum_{w \in \pi_t^W} b_{wt}}{\sum_{t \in T} \sum_{s \in \pi_t^S} a_{st} + \sum_{t \in T} \sum_{w \in \pi_t^W} b_{wt}}. \tag{29}$$

The first three R-D (resource destruction) operators can be considered the traditional worst removal operators, which improve the objective function by removing high-cost assignments. However, these R-D operators may be shortsighted for large-sized problems that are difficult to solve. To address this issue, we also employ the T-D (target destruction) operators to expand the neighborhood, which removes all the assigned resources (weapons and sensors) for a set of chosen targets. In general, the T-D operators always destroy more assignments than the R-D operators because the R-D operators only remove δ resources, while the T-D operators remove δ targets, each of which has at least one resource. Therefore, these two types of destruction operators mainly differ in the level of destruction. Their performance is evaluated in the numerical experiments in Section VI-D.

C. Adaptive selection mechanism

Given that the destruction phase employs six destruction operators, it thus needs to decide which operator should be selected for each iteration. We use the adaptive mechanism proposed in Ropke *et al*. [34] to assign each destruction operator a probability of being selected. The probability of operator θ being selected is computed as
$$prob(\theta) = \frac{\Omega^\theta}{\sum_{\theta' \in D} \Omega^{\theta'}}, \tag{30}$$
where the weights Ω of all operators are initialized to the same value η and updated based on the performance of these operators. Specifically, the chosen destruction operator for each iteration is given a score based on its performance in the repair phase, depending on whether *a new best solution*, *a feasible solution*, or *no feasible solution* is discovered (See Section V-D). We set the basic scores for the three cases as v_1, v_2, v_3, with $v_1 > v_2 > v_3$. The weight is not updated for each iteration to avoid some operators having excessively high or low weights. Instead, the weights Ω are updated every 10 iterations based on the cumulative scores computed by
$$\Omega^\theta = \lambda \Omega^\theta + (1-\lambda) \frac{\xi^\theta}{\max\{\omega^\theta, 1\}}, \tag{31}$$
where ξ^θ is the cumulative score of operator θ in the current 10 iterations, ω^θ is the number of times the operator θ has been called, and λ is the discount coefficient.

D. Repair phase

The destroyed solution is subjected to the repair phase to retain feasibility. This is achieved by constructing and solving a mixed integer programming model. The fundamental idea of this model-driven repair phase is to construct a subproblem by optimizing the destroyed part while keeping the rest of the solution fixed, making it solvable by a general-purpose solver. Three types of subproblems are generated based on the resources involved in the destruction operators. For the destruction operators that handle sensors (S-D or TS-D), the subproblem SP1 only considers the sensors not assigned to any target and the targets with unsatisfied destruction probability. For the destruction operators that only handle weapons (W-D or TW-D), the subproblem SP2 only considers the weapons not assigned to any target and the targets with unsatisfied destruction probability. For the destruction operators that handle both sensors and weapons (SW-D or TSW-D), the subproblem SP3 considers both the sensors and weapons not assigned to any target and the targets with unsatisfied destruction probability.

Specifically, the subproblem SP1 can be formulated as follows:

$$\text{(SP1)} \ \min \sum_{i\in S'}\sum_{k\in T'} a_{ik}x_{ik} \tag{32}$$

$$\text{s.t.} \ \sum_{k\in T'} x_{ik} \le 1, \ \forall i \in S', \tag{33}$$

$$\sum_{i\in S'} x_{ik} \ge 1, \ \forall k \in T', \tag{34}$$

$$\Phi_k^W(1-\Phi_k^S \prod_{i\in S'}(1-p_{ik})^{x_{ik}}) \ge u, \ \forall k \in T', \tag{35}$$

$$x_{ik} \in \{0,1\}, \ \forall i \in S', \ k \in T', \tag{36}$$

where $S' = S/\pi^S$ denotes the set of unassigned sensors, and T' denotes the set of targets with unsatisfied desired probability, and Φ_k^S is the destruction probability contributed by the sensors not removed for target k. As SP1 does not consider the assignment of weapons to targets, i.e., the values of the involved variables are fixed at the same values as the reference solution, we denote $1-\prod_{y\in W}(1-q_{jk})^{y_{jk}}$ of constraint (6) in the original problem as a constant Φ_k^W for each target k. The objective is to minimize the total assignment cost of the sensors. Note that Eq. (35) can be linearized as

$$\sum_{i\in S'} \ln(1-p_{ik})x_{ik} \le \ln \frac{1-\frac{u}{\Phi_k^W}}{\Phi_k^S}, \ \forall k \in T'. \tag{35'}$$

Similarly, SP2 can be formulated as:

$$\text{(SP2)} \ \min \sum_{j\in W'}\sum_{k\in T'} b_{jk}y_{jk} \tag{37}$$

$$\text{s.t.} \ \sum_{k\in T'} y_{jk} \le 1, \ \forall j \in W', \tag{38}$$

$$\sum_{j\in W'} y_{jk} \ge 1, \ \forall k \in T', \tag{39}$$

$$\Phi_k^S(1-\Phi_k^W \prod_{j\in W'}(1-q_{jk})^{y_{jk}}) \ge u, \ \forall k \in T', \tag{40}$$

$$y_{jk} \in \{0,1\}, \ \forall j \in W', \ k \in T'. \tag{41}$$

where the nonlinear constraint (40) can be restate as:

$$\sum_{j\in W'} \ln(1-q_{jk})y_{jk} \le \ln \frac{1-\frac{u}{\Phi_k^S}}{\Phi_k^W}, \ \forall k \in T'. \tag{40'}$$

and the subproblem SP3 can be formulated as follows:

$$\text{(SP3)} \ \min \sum_{i\in S'}\sum_{k\in T'} a_{ik}x_{ik} + \sum_{j\in W'}\sum_{k\in T'} b_{jk}y_{jk} \tag{42}$$

$$\text{s.t.} \ \sum_{k\in T'} x_{ik} \le 1, \ \forall i \in S' \tag{43}$$

$$\sum_{k\in T'} y_{jk} \le 1, \ \forall j \in W', \tag{44}$$

$$\left(1-\Phi_k^S \prod_{i\in S'}(1-p_{ik})^{x_{ik}}\right)\left(1-\Phi_k^W \prod_{j\in W'}(1-q_{jk})^{y_{jk}}\right) \ge u, \ \forall k \in T', \tag{45}$$

$$x_{ik} \in \{0,1\}, \ \forall i \in S', \ k \in T', \tag{46}$$

$$y_{jk} \in \{0,1\}, \ \forall j \in W', \ k \in T'. \tag{47}$$

Constraint (45) can be linearized using the method introduced in Section IV-B.

Note that due to the potential hardness of the subproblems when certain variables are fixed, the repair phase may not produce a feasible solution within the given time limit μ. Nonetheless, the adaptive mechanism ensures that more feasible solutions are generated as far as possible by assigning a lower score to such a destruction operator.

VI. Computational results

In this section, we conduct computational experiments to evaluate the performance of the linear approximations for lower and upper bounds and the adaptive large neighborhood search algorithm. Moreover, the effectiveness of the model-driven repair phase and the adaptive operator selection mechanism is analyzed through controlled experiments.

A. Instances and computational setups

We generate a set of 30 instances by referring to Xin *et al.* [6], which can be downloaded at https://github.com/NWPU-ORMS/SWTA. Specifically, a_{ik} and b_{jk} are sampled from $[1, 1000]$, p_{ik} from $[0.50, 0.95]$, and q_{jk} from $[0.72, 0.96]$, uniformly. Our preliminary experiments find that small instances with no more than 40 sensors and 40 weapons can be solved to optimality by both the Gurobi solver and our ALNS algorithm within no more than one minute. Hence, the results for these instances are not reported. For each combination of the numbers of sensors, weapons and targets, we generate 10 independent instances, where the smallest instance is of size S60-W60-T40 and the largest is of size S500-W500-T360. In addition, the minimum desired probability for target destruction is $u \in \{0.8, 0.85, 0.9\}$. Hence, we obtained 30 differentiated instances that are used in our experiments.

Our ALNS algorithm was coded in the C++ programming language and compiled with the GCC 10.2.0 compiler. Table I lists the parameter settings of ALNS. Preliminary experiments indicate that when the weight γ is biased towards the cost, the quality of the initial solution is usually better. However, we may have a high chance of obtaining an infeasible solution. Hence, we set $\gamma = 10^{-3}$ for instances with $u = 0.8$ since the constraint (6) is easily satisfied. For other instances, we set $\gamma = 10^{-4}$ or 10^{-5}. For the destruction degree δ, we set $0.2 \times (S+W)$ for instances starting from S60-W60-T40 to S250-W250-T160 and $0.1 \times (S+W)$ for other instances. Moreover, we set the time limit $\mu = 1$ second for solving each subproblem. The weight of the operators is discounted by $\lambda = 0.95$ after a certain number of iterations. All experiments are conducted on a computing cluster equipped with dual Intel® Xeon® Gold 6226R CPU and 256 GB RAM on each node. The approximation models, (LB) and (UB), and the subproblem models, (SP1)–(SP3), are solved using the Gurobi 9.0.3 solver.

TABLE I
SETTINGS OF PARAMETERS FOR ALNS

Parameters	Section	Description	Values
γ	V-A	trade-off coefficient of cost and probability in the initial solution	$10^{-3}, 10^{-4}, 10^{-5}$
δ	V-B	number of elements selected to destroy	$\{0.1, 0.2\}\times$(S+W)
η	V-C	weight of destruction operators at the beginning of the algorithm	1
v_1, v_2, v_3	V-C	score for destruction operator in each iteration	4, 2, 0
λ	V-C	discount coefficient of the last weight	0.95
μ	V-D	time limit(s) for optimization in each iteration	1

B. Evaluation of lower and upper bounds

The approximation models (LB) and (UB) with many decomposed parts n may yield tighter bounds but are harder to solve. In the experiments, we set seven different levels of $n \in \{20, 50, 100, 200, 300, 500, 1000\}$. For each instance and each value of n, we set a time limit of 10 hours and record the obtained best lower and upper bounds and the computational time.

Table II presents the lower and upper bounds obtained for each instance. The column "Obj" represents the optimal objective value of the respective model. Note that "-" means that the optimal solution of the corresponding models (LB) and (UB) cannot be found within the time limit, which also indicates that no valid bounds can be found for these instances at given n. This table shows that the approximation method can obtain lower and upper bounds for 18 instances but fails for others. Moreover, the lower and upper bounds are equal for 10 instances, indicating that these solution values are optimal for S-WTA. Although the gaps between the lower and upper bounds, calculated as $(UB - LB)/LB \times 100\%$, are generally less than 2%, the runtime of the Gurobi solver increases significantly as the instance size increases. In conclusion, this experiment shows that our proposed approximation methods can obtain good bounds for most instances.

Table III presents detailed results of using different n for the 10 instances with proved optimal solutions. For each instance we increase the value of n from 20 to 1000 and solve the approximation models (LB) and (UB) to reach optimal solutions of the models. We mark in bold such a pair of lower and upper bounds that hits the optimal value first. Let's use S60-W60-T40 with $u = 0.8$ as an example. The optimal value 4110 is obtained by setting $n = 20$ to solve the approximation model (UB) and setting $n = 1000$ to solve (LB). For this instance, we prove its optimal objective value to be 4110 until we increase n to 1000.

The results in Table III reveal that solving the approximation models can obtain proven optimal solutions for 7 instances when $n = 1000$ and 500 and for 3 instances when $n = 300, 200, 100$, respectively. As n increases, the proven optimal solutions can be found for more instances. However, large values of n lead to an increased number of binary variables z_{kl} in models (LB) and (UB). And the required time for solving the approximation models grows as the value of n increases. Hence, both upper and lower bounds can be improved by progressively increasing n, but there's a possibility of encountering instances where solving time exceeds the given limit. When determining an appropriate setting of n, a trade-off exists between the solution quality obtained and the time taken to solve the resulting models. In our experiments, increasing n to 1000 can achieve valid bounds that are either proven optimal or exhibit a tight gap of less than 2% for 18 out of 30 instances. However, for the other 12 instances, obtaining valid bounds within a long time limit is challenging. Therefore, the experiments for computing lower and upper bounds are terminated at $n = 1000$.

C. Comparisons of different algorithms

To the best of our knowledge, no study has addressed the minimum desired probability of target destruction when conducting the joint allocation of weapons and sensors. We note that Kwon *et al.* [4], [8] investigated a WTA problem that shares a similar objective and nonlinear constraint with our S-WTA. However, expanding their methods to address the S-WTA problem is not straightforward due to inherent differences in problem structures and constraints. In their problem, the left-hand side of the nonlinear constraint comprises a constant and a product term. This structure allows for the linearization of the nonlinear constraint through logarithmic transformation. However, in our S-WTA problem, the left-hand side of the nonlinear constraint consists of a product term involving two factors, and each of these factors resembles the product term observed in the WTA problem. Thus, a direct application of logarithmic transformation is not feasible because of the nesting product terms in this scenario. To assess the performance of the ALNS algorithm, we compare it with the following four reference methods:

- UB model: It solves the model (UB) presented in section IV by Gurobi directly. From Table II, 9 out of 18 instances obtained the best solution of (UB) with $n = 100$ using the Gurobi solver. Hence, we opt to set $n = 100$ for this method.
- ALNS-H: It replaces the model-driven repair phase in ALNS with heuristic repair operators, including greedy and regret repair operators. The greedy repair operators assign sensors or weapons by combining multiple greedy rules, including the lower cost rule, higher probability rule, and a lower ratio of cost/probability rule. The regret repair operators give each infeasible target a regret value. Suppose $C(t) = \{C_1, C_2, \ldots, C_k\}$ is the sorted set of costs by assigning sensors or weapons to target t, the regret value is defined as $\sum_{i=1}^{k}(C_i - C_1)$, and we set $k = 2, 3, 4$ for different regret operators. Resources with lower costs will be assigned to targets with higher regret values at an earlier stage.
- MRBCH: The algorithm was proposed by Xin *et al.* [6] for solving the S-WTA problem with target threat value minimization. The fundamental idea behind this constructive heuristic involves iteratively assigning appropriate weapons or sensors to targets based on their marginal return. Here, we adopt this algorithm by modifying the way of computing the marginal return. Specifically, our study additionally consider the assignment costs associated with weapons and sensors, a factor that was overlooked in the original paper. Hence, we calculate the marginal return of each potential assignment (s, w, t) as $p \times (1000 - c)$,

TABLE II
LOWER AND UPPER BOUNDS OF S-WTA WITH $36000s$

Instances				LB			UB			$\frac{UB-LB}{LB}(\%)$
S	W	T	u	Obj	Time(s)	n	Obj	Time(s)	n	
60	60	40	0.80	4110	1039	1000	4110*	25	20	0.00
			0.85	5740	2781	500	5740*	1260	300	0.00
			0.90	8875	1771	100	8875*	470	20	0.00
90	90	60	0.80	4040	8105	1000	4040*	686	100	0.00
			0.85	5211	6150	500	5240	1255	100	0.56
			0.90	8064	11539	200	8064*	4201	100	0.00
120	120	80	0.80	3997	9405	1000	3997*	765	100	0.00
			0.85	5235	2000	100	5235*	7499	500	0.00
			0.90	7654	33847	300	7654*	8115	50	0.00
150	150	100	0.80	4284	35572	1000	4284*	545	20	0.00
			0.85	5197	7590	200	5204	3355	100	0.13
			0.90	7657	21163	20	-	-	-	-
200	200	120	0.80	3255	27028	1000	3255*	1538	100	0.00
			0.85	4178	21481	500	4183	10295	300	0.12
			0.90	5663	18300	100	5731	28246	100	1.20
250	250	160	0.80	3802	6728	200	3818	1136	20	0.42
			0.85	4847	25086	200	4865	12711	100	0.37
			0.90	-	-	-	-	-	-	-
300	300	200	0.80	4125	18136	200	4143	10985	100	0.43
			0.85	5314	30948	50	-	-	-	-
			0.90	-	-	-	-	-	-	-
350	350	250	0.80	4668	24841	50	4758	24562	50	1.93
			0.85	-	-	-	-	-	-	-
			0.90	-	-	-	-	-	-	-
400	400	300	0.80	-	-	-	-	-	-	-
			0.85	-	-	-	-	-	-	-
			0.90	-	-	-	-	-	-	-
500	500	360	0.80	-	-	-	-	-	-	-
			0.85	-	-	-	-	-	-	-
			0.90	-	-	-	-	-	-	-

Note: UB with $*$ denotes the optimal value

TABLE III
DETAILED RESULTS OF BOUNDS WITH DIFFERENT n FOR PROVEN OPTIMAL INSTANCES

Instances				$n=20$		50		100		200		300		500		1000	
S	W	T	u	LB	UB	LB	UB	LB	UB	LB	UB	LB	UB	LB	UB	LB	UB
60	60	40	0.80	3929	**4110**	3999	4114	4019	4110	4094	4110	4094	4110	4094	4110	**4110**	4110
60	60	40	0.85	5538	5759	5687	5742	5716	5742	5720	5742	5720	**5740**	**5740**	5742	5740	5740
60	60	40	0.90	8771	**8875**	8794	8875	**8875**	8875	8875	8875	8875	8875	8875	8875	8875	8875
90	90	60	0.80	3802	4071	3934	4060	3958	**4040**	4021	4040	4021	4040	4021	4040	**4040**	4040
90	90	60	0.90	7779	8167	7898	**8064**	8005	8064	**8064**	8064	8064	8064	8064	8064	-	8237
120	120	80	0.80	3891	4000	3947	4000	3971	**3997**	3971	3997	3971	3997	3971	3997	**3997**	3997
120	120	80	0.85	5049	5268	5203	5245	**5235**	5244	5235	5244	5235	5244	5235	**5235**	5235	5235
120	120	80	0.90	7472	7699	7589	**7654**	7618	7654	7618	7654	**7654**	7654	-	7659	-	-
150	150	100	0.80	4098	**4284**	4180	4307	4210	4284	4255	4284	4255	4284	4255	4284	**4284**	4284
200	200	120	0.80	3181	3290	3227	3260	3240	**3255**	3243	3255	3250	3255	3254	3255	**3255**	3255

where p and c denote the increased destruction probability and increased objective value brought by this assignment. For each iteration the assignment with the highest marginal return is executed. By adopting this strategy, the resulting solution is more likely to be feasible and obtains a better objective value.

- VLSN: The adapted implementation of the VLSN algorithm, also called ejection chain algorithm, follows the core concept proposed by Ahuja *et al*. [9]. The algorithmic framework is tailored to accommodate the joint assignment of sensors and weapons. Initially, the solution is generated like ALNS. Subsequently, we initialize the chain length to 1 and set the longest length to 3. During the search, the assignment scheme of sensors/weapons remains fixed, while the remaining execute cyclic exchange with length k. If no better solution is identified, k is incremented by one. If k exceeds the longest length, we fix the current solution of weapons/sensors and continue the search in sensors/weapons. The algorithm terminates when no improved solution can be found in any neighborhood.

We first analyze the running profile of the ALNS algorithm. Figure 2 shows how the objective value varies as the iteration increases on the instance S60-W60-T40 with different destruction probabilities $u = 0.80, 0.85, 0.90$. The horizontal axis represents the number of iterations, while the vertical axis denotes the gap between the best objective value obtained at the given iteration to the optimal objective

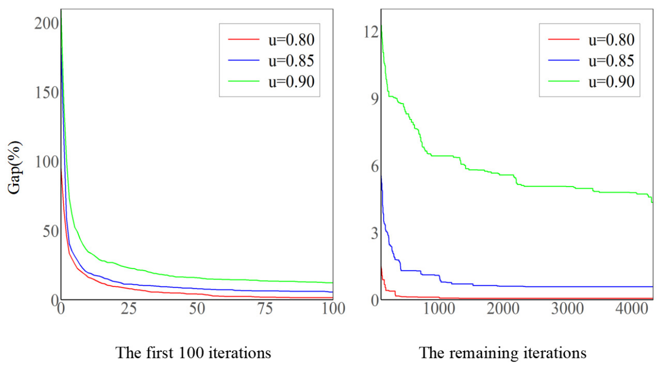


Fig. 2. The convergence process of instance S60-W60-T40.

value. We can observe that in the first 100 iterations, the gap dramatically decreases from 100% to 2%, 185% to 6% and 210% to 15% under $u = 0.80, 0.85, 0.90$, respectively. On the contrary, the gap can further decrease and converges to optimal or suboptimal solution in the following large number of iterations. The same convergence trend can be found for the other instances. This experimental analysis indicates that our algorithm presents good search performance.

Furthermore, we conduct an additional experiment using a running time of 300 seconds to compare the proposed ALNS algorithm with four reference methods. For instances where the optimal solution is known, the execution of each algorithm is terminated upon reaching the optimal values. We record the average objective value found by each algorithm and the computational time required to attain this value. In addition, we record the objective percentage gap between ALNS and the other four algorithms, calculated by $(\text{Obj}^{\text{A}} - \text{Obj}^{\text{ALNS}})/\text{Obj}^{\text{ALNS}} \times 100$, where Obj^{A} is the objective value of the comparison algorithms. A positive value of $gap\%$ in columns "UB model", "ALNS-H", "MRBCH" and "VLSN" means they are worse than ALNS. For a detailed analysis, we divide the 30 instances into small and large groups according to the instance size. The small group includes instances with no more than 200 sensors/weapons, while the large group includes the other instances. The detailed experimental results are presented in online supplement.

TABLE IV
COMPARISON RESULTS OF ALNS WITH FOUR REFERENCE METHODS

Compared Methods	Instances		Destruction Probability(u)			p–value
	Small	Large	0.80	0.85	0.90	
UB model	4	11	2	5	8	0.005745
ALNS-H	15	15	10	10	10	9.127×10^{-7}
MRBCH	15	15	10	10	10	7.151×10^{-5}
VLSN	15	15	10	10	10	9.127×10^{-7}

Table IV presents the summarized results by comparing ALNS with the reference methods. Each column under "Instances" or "Destruction Probability(u)" records the count of instances where ALNS outperformed the compared method. For the small instances, we see that ALNS outperforms the UB model for 4 out of 15 instances. For the instances where ALNS performs worse, the objective gaps to the UB results are generally smaller than 2%. This outcome demonstrates that the performance of ALNS is competitive with that of the UB model for solving small instances. For the large instances, ALNS performs much better for 11 out of 15 instances. Especially for instances S350-W350-T250 to S500-W500-T360 with $u = 0.9$, the objective gaps are more than 100%. The superiority over the UB model is significantly better by obtaining a p-value of 0.005745.

Compared with the variant ALNS-H, our proposed ALNS algorithm performs significantly better by obtaining a p-value of 9.127×10^{-7} on all 30 instances. In addition, ALNS-H achieves better results than the UB model for large instances with $u = 0.9$. This indicates that the Gurobi solver cannot obtain acceptable solutions for large instances with high desired destruction probability within a reasonable running time.

In addition, our ALNS algorithm outperforms the reference algorithms MRBCH and VLSN. Although MRBCH and VLSN obtain their final solutions rapidly, the average gaps of MRBCH and VLSN are quite large. Moreover, wilcoxon testings in terms of solution quality indicate that our ALNS algorithm is significantly better than MRBCH and VLSN by obtaining the p-values of 7.151×10^{-5} and 9.127×10^{-7}, respectively. In particular, MRBCH cannot yield feasible solutions for all instances with $u = 0.9$. A similar observation applies to the comparison between VLSN and ALNS, where the former has higher gaps for instances with $u = 0.9$. This analysis highlights the challenge associated with the nonlinear constraints in our S-WTA problem, particularly when dealing with higher probabilities of destruction.

D. Analysis of destruction strategies and adaptive mechanism

Recall that the destruction phase of ALNS adaptively selects a destruction operator among six destruction operators by an adaptive mechanism. In this subsection, we first analyze the role of different destruction operators in solving instances. During the ALNS execution, we additionally collect the number of times each of the six destruction operators updated the best solution, represented by $\{d_1, d_2, \ldots, d_6\}$. Then, we summarize the updating frequency of the ith destruction operator by $F_i = d_i / \sum_{i=1}^{6} d_i$. Table V presents the frequency of updating the best solution for each destruction operator in the ALNS execution.

From Table V, we find that the three operators based on target-destruction, i.e., TS-D, TW-D, TSW-D, obtain the sum average frequencies of 0.64, 0.68, and 0.73 when $u = 0.8$, $u = 0.85$, $u = 0.9$, respectively. This indicates that target destruction operators are more important than resource destruction operators in ALNS.

On the other hand, the frequency of each destruction operator varies for different instances. For example, the destruction operator TSW-D performs much better for instances with a required small destruction probability u. Conversely, the destruction operators TS-D and TW-D get an upward trend in frequency to update the best solution for instances with $u = 0.8$ to $u = 0.9$. Due to the complexity of the subproblem SP3 for solving instances with a large u, the corresponding destruction operator TSW-D cannot obtain improving solutions frequently. In addition, our problem instances have the

TABLE V
FREQUENCY OF UPDATING THE BEST SOLUTION FOR EACH DESTRUCTION OPERATOR

Instances				Destruction operators					
S	W	T	u	S-D	W-D	TS-D	TW-D	SW-D	TSW-D
60	60	40	0.80	0.09	0.07	0.07	0.10	0.21	0.47
			0.85	0.06	0.09	0.17	0.22	0.10	0.36
			0.90	0.05	0.13	0.25	0.32	0.08	0.16
90	90	60	0.80	0.04	0.11	0.13	0.15	0.17	0.40
			0.85	0.06	0.08	0.23	0.27	0.12	0.24
			0.90	0.06	0.10	0.35	0.31	0.07	0.12
120	120	80	0.80	0.09	0.08	0.17	0.12	0.15	0.39
			0.85	0.07	0.07	0.23	0.26	0.16	0.20
			0.90	0.06	0.08	0.27	0.35	0.10	0.14
150	150	100	0.80	0.05	0.11	0.18	0.20	0.17	0.29
			0.85	0.08	0.11	0.20	0.21	0.16	0.22
			0.90	0.06	0.12	0.28	0.33	0.09	0.12
200	200	120	0.80	0.09	0.09	0.16	0.16	0.21	0.29
			0.85	0.07	0.10	0.16	0.24	0.16	0.27
			0.90	0.06	0.10	0.23	0.28	0.16	0.17
250	250	160	0.80	0.08	0.11	0.18	0.14	0.20	0.29
			0.85	0.06	0.11	0.19	0.25	0.14	0.24
			0.90	0.12	0.12	0.28	0.37	0.10	0.01
300	300	200	0.80	0.09	0.11	0.13	0.15	0.16	0.37
			0.85	0.08	0.14	0.18	0.24	0.13	0.24
			0.90	0.07	0.09	0.25	0.34	0.10	0.15
350	350	250	0.80	0.09	0.10	0.22	0.16	0.14	0.30
			0.85	0.08	0.13	0.19	0.23	0.15	0.22
			0.90	0.07	0.10	0.30	0.37	0.06	0.10
400	400	300	0.80	0.10	0.11	0.16	0.21	0.20	0.21
			0.85	0.09	0.11	0.24	0.27	0.14	0.15
			0.90	0.10	0.09	0.34	0.39	0.04	0.04
500	500	360	0.80	0.12	0.14	0.19	0.18	0.18	0.20
			0.85	0.09	0.12	0.23	0.26	0.15	0.15
			0.90	0.09	0.12	0.32	0.36	0.07	0.03

sensor detection probability $p_{ik} \in [0.50, 0.95]$ and weapon destruction probability $q_{jk} \in [0.72, 0.96]$. The higher skew of weapons over sensors suggests that the assignment of weapons has a more significant impact on the solution quality. Hence, sensor-related destruction operators such as S-D and TS-D are slightly inferior to weapon-related operators W-D and TW-D.

Furthermore, we analyze whether incorporating the adaptive mechanism plays a crucial role in ALNS. To perform this experiment, we generate an ALNS variant by replacing the adaptive mechanism with a random strategy for selecting destruction operators. We can achieve the same and unchangeable weight of selecting each operator by simply setting v_1, v_2, v_3 to 0 and λ to 1. We rerun the experiment for this ALNS variant using the same experimental setting as in Section VI-C. Figure 3 shows the comparison between the ALNS algorithms with and without the adaptive mechanism. In this figure, the horizontal axis represents instances of different sizes, and the vertical axis records the objective percentage gaps between the two algorithm versions for each instance, calculated by $(\text{Obj}_{\text{with}} - \text{Obj}_{\text{w\textbackslash o}})/\text{Obj}_{\text{w\textbackslash o}} \times 100$.

The negative value of $gap\%$ means that a better objective value is obtained by the ALNS algorithm with the adaptive mechanism for solving this instance, and vice versa. We observe from Figure 3 that the ALNS algorithm with the adaptive mechanism performs better for 22 out of 30 instances when compared to the one without the adaptive mechanism. In addition, the adaptive mechanism is more important when solving large instances and particularly effective when $u = 0.9$. This observation is consistent with the analysis in Table V. For large instances, i.e., S350-W350-T250 to S500-W500-T360, the use frequencies of six operators vary between $[0.09, 0.30]$, $[0.10, 0.21]$ and $[0.12, 0.20]$ when $u = 0.8$, whereas these values vary between $[0.06, 0.37]$, $[0.04, 0.39]$ and $[0.03, 0.36]$ when $u = 0.9$. This means that for instances of large size and desired destruction probability, the ALNS algorithm has a more obvious preference for operators. Thus, the adaptive mechanism is fundamental.

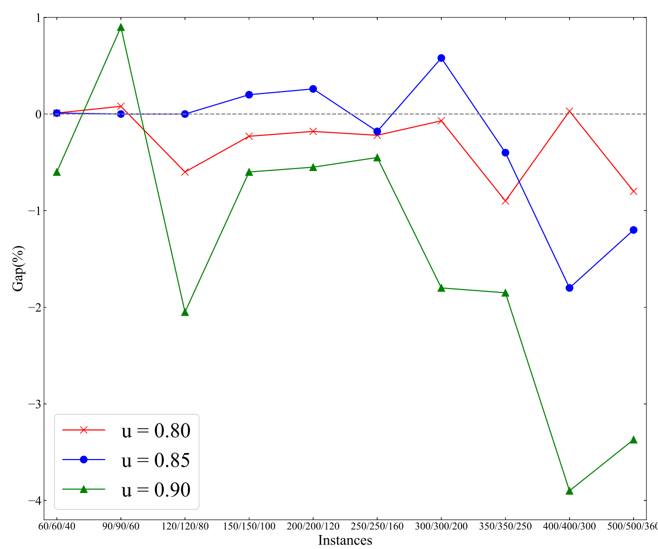


Fig. 3. ALNS comparison: with and without adaptive mechanism.

VII. CONCLUSION

In this paper, we studied a novel S-WTA problem aimed at minimizing assignment cost while ensuring minimum desired destruction probability for each target. We formulated this problem as an integer nonlinear programming model and derived the lower and upper bounds by constructing its linear approximations. To solve this problem, we propose an adaptive large neighborhood search heuristic composed of a destruction phase and a model-driven repair phase. We conducted numerical experiments on various instance sets to test the proposed models and algorithms. The results indicate that the derived lower and upper bounds are quite close for small instances. Our ALNS algorithm exhibits competitive performance with the Gurobi solver for small instances while demonstrating superior effectiveness for large instances. Furthermore, controlled experiments show that our model-driven repair phase outperforms traditional heuristic repair operators. In comparison with the reference methods that have been adapted from the literature, the ALNS algorithm also shows a significant strength. For practical implementation in actual combat scenarios, once the commander has collected sufficient battlefield information and established the desired levels of destruction based on various considerations, our ALNS algorithm can be used to ensure efficient coordination of diverse combat resources with reduced operational costs.

In future research, dynamic S-WTA problems can be studied to simulate more realistic battlefield situations with increasing combat resources and targets over time. Additionally, given the high demand for rapid responses, investigating effective neighborhood structures along with acceleration techniques would help improve the computational efficiency of our algorithm. Since the linearization method of approximation models is computationally intensive, future research could explore exact algorithms such as branch-and-cut or column generation.

References

[1] A. S. Manne, "A target-assignment problem," *Operations Research*, vol. 6, no. 3, pp. 346–351, 1958.

[2] S. P. Lloyd and H. S. Witsenhausen, "Weapons allocation is np-complete," in *1986 Summer Computer Simulation Conference*, 1986, pp. 1054–1058.

[3] G. denBroeder Jr, R. Ellison, and L. Emerling, "On optimum target assignments," *Operations Research*, vol. 7, no. 3, pp. 322–326, 1959.

[4] O. Kwon, K. Lee, D. Kang, and S. Park, "A branch-and-price algorithm for a targeting problem," *Naval Research Logistics*, vol. 54, no. 7, pp. 732–741, 2007.

[5] C. N. Bogdanowicz, ZR, "Sensor-target and weapon-target pairings based on auction algorithm," in *Proceedings of the 11th WSEAS International Conference on Applied Mathematics*. Dallas, TX, USA, 2007, pp. 22–24.

[6] B. Xin, Y. Wang, and J. Chen, "An efficient marginal-return-based constructive heuristic to solve the sensor–weapon–target assignment problem," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 12, pp. 2536–2547, 2018.

[7] X. Li, D. Zhou, Z. Yang, Q. Pan, and J. Huang, "A novel genetic algorithm for the synthetical sensor-weapon-target assignment problem," *Applied Sciences*, vol. 9, no. 18, p. 3803, 2019.

[8] O. Kwon, D. Kang, K. Lee, and S. Park, "Lagrangian relaxation approach to the targeting problem," *Naval Research Logistics*, vol. 46, no. 6, pp. 640–653, 1999.

[9] R. K. Ahuja, A. Kumar, K. C. Jha, and J. B. Orlin, "Exact and heuristic algorithms for the weapon-target assignment problem," *Operations Research*, vol. 55, no. 6, pp. 1136–1146, 2007.

[10] A. G. Kline, D. K. Ahner, and B. J. Lunday, "Real-time heuristic algorithms for the static weapon target assignment problem," *Journal of Heuristics*, vol. 25, pp. 377–397, 2019.

[11] Y. Lu and D. Z. Chen, "A new exact algorithm for the weapon-target assignment problem," *Omega*, vol. 98, p. 102138, 2021.

[12] A. C. Andersen, K. Pavlikov, and T. A. Toffolo, "Weapon-target assignment problem: Exact and approximate solution algorithms," *Annals of Operations Research*, vol. 312, pp. 581–606, 2022.

[13] G. Shang, Z. Zhang, X. Zhang, and C. Cao, "Immune genetic algorithm for weapon-target assignment problem," in *Workshop on Intelligent Information Technology Application (IITA 2007)*. IEEE, 2007, pp. 145–148.

[14] E. Sonuc, B. Sen, and S. Bayir, "A parallel simulated annealing algorithm for weapon-target assignment problem," *International Journal of Advanced Computer Science and Applications*, vol. 8(4), pp. 87–92, 2017.

[15] A. G. Kline, D. K. Ahner, and B. J. Lunday, "A heuristic and metaheuristic approach to the static weapon target assignment problem," *Journal of Global Optimization*, vol. 78, pp. 791–812, 2020.

[16] W. Luo, J. Lü, K. Liu, and L. Chen, "Learning-based policy optimization for adversarial missile-target assignment," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 7, pp. 4426–4437, 2021.

[17] S. Li, X. He, X. Xu, T. Zhao, C. Song, and J. Li, "Weapon-target assignment strategy in joint combat decision-making based on multi-head deep reinforcement learning," *IEEE Access*, 2023.

[18] M. Cao and W. Fang, "Swarm intelligence algorithms for weapon-target assignment in a multilayer defense scenario: A comparative study," *Symmetry*, vol. 12, no. 5, p. 824, 2020.

[19] J. Zhang, M. Kong, G. Zhang, and Y. Huang, "Weapon–target assignment using a whale optimization algorithm," *International Journal of Computational Intelligence Systems*, vol. 16, no. 1, p. 62, 2023.

[20] X. Wu, C. Chen, and S. Ding, "A modified moea/d algorithm for solving bi-objective multi-stage weapon-target assignment problem," *IEEE Access*, vol. 9, pp. 71 832–71 848, 2021.

[21] J. Huang, X. Li, Z. Yang, W. Kong, Y. Zhao, and D. Zhou, "A novel elitism co-evolutionary algorithm for antagonistic weapon-target assignment," *IEEE Access*, vol. 9, pp. 139 668–139 684, 2021.

[22] E. Acar, S. Hatipoğlu, and İ. Yılmaz, "A quantum algorithm for solving weapon target assignment problem," *Engineering Applications of Artificial Intelligence*, vol. 125, p. 106668, 2023.

[23] M. S. Hughes and B. J. Lunday, "The weapon target assignment problem: Rational inference of adversary target utility valuations from observed solutions," *Omega*, vol. 107, p. 102562, 2022.

[24] Y. Ma, G. Wang, X. Hu, and H. Luo, "Two-stage hybrid heuristic search algorithm for novel weapon target assignment problems," *Computers & Industrial Engineering*, vol. 162, p. 107717, 2021.

[25] T. Wang, L. Fu, Z. Wei, Y. Zhou, and S. Gao, "Unmanned ground weapon target assignment based on deep q-learning network with an improved multi-objective artificial bee colony algorithm," *Engineering Applications of Artificial Intelligence*, vol. 117, p. 105612, 2023.

[26] A. Kline, D. Ahner, and R. Hill, "The weapon-target assignment problem," *Computers & Operations Research*, vol. 105, pp. 226–236, 2019.

[27] Z. R. Bogdanowicz, "A new efficient algorithm for optimal assignment of smart weapons to targets," *Computers & Mathematics with Applications*, vol. 58, no. 10, pp. 1965–1969, 2009.

[28] Z.-F. Li, X.-M. Li, J.-J. Dai, J.-Z. Chen, and F.-X. Zhang, "Sensor-weapon-target assignment based on improved swt-opt algorithm," in *2011 IEEE 2nd International Conference on Computing, Control and Industrial Engineering*, vol. 2. IEEE, 2011, pp. 25–28.

[29] H. Chen, Z. Liu, Y. Sun, and Y. Li, "Particle swarm optimization based on genetic operators for sensor-weapon-target assignment," in *2012 Fifth International Symposium on Computational Intelligence and Design*, vol. 2. IEEE, 2012, pp. 170–173.

[30] K. Zhang, D. Zhou, Z. Yang, W. Kong, and L. Zeng, "A novel heterogeneous sensor-weapon-target cooperative assignment for ground-to-air defense by efficient evolutionary approaches," *IEEE Access*, vol. 8, pp. 227 373–227 398, 2020.

[31] Y. Zhou, J.-K. Hao, and B. Duval, "Frequent pattern-based search: A case study on the quadratic assignment problem," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 3, pp. 1503–1515, 2020.

[32] Y. Jin and J.-K. Hao, "Effective learning-based hybrid search for bandwidth coloring," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 4, pp. 624–635, 2014.

[33] A. N. Gullhav, J.-F. Cordeau, L. M. Hvattum, and B. Nygreen, "Adaptive large neighborhood search heuristics for multi-tier service deployment problems in clouds," *European Journal of Operational Research*, vol. 259, no. 3, pp. 829–846, 2017.

[34] S. Ropke and D. Pisinger, "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows," *Transportation Science*, vol. 40, no. 4, pp. 455–472, 2006.

Online supplement of "Efficient Adaptive Large Neighborhood Search for Sensor–Weapon–Target Assignment"

Yang Wang, Junpeng Wang, Jin-Kao Hao, Jianguang Feng

I. Lower and upper bounds

A. The proof of Proposition 1

Let $\mathbb{P}$ and $\mathbb{P}_{\mathrm{L}}$ be the solution space of (S-WTA) and (LB), and obj and obj_l be their respective optimal objective values. For any feasible solution (A_k, B_k) of (S-WTA), one particular l can be found that satisfies

$$a_l \geq A_k \geq a_{l-1}.$$

Due to $A_k B_k \geq u$, we have

$$B_k \geq \frac{u}{A_k} \geq \frac{u}{a_l} = b_l.$$

Therefore, if A_k and B_k satisfy constraint (6), there must exist some $z_{kl} = 1$ such that constraints (9)–(12) hold. In other words, any feasible solution of (S-WTA) is also feasible for (LB). As a result, we have $\mathbb{P}_{\mathrm{L}} \supseteq \mathbb{P}$. Because formulations (S-WTA) and (LB) have the same objective and other constraints, it can be concluded that $obj_l \leq obj$. Thus, the optimal objective value of (LB) is a valid lower bound of (S-WTA).

B. The proof of Proposition 2

Let $\mathbb{P}$ and $\mathbb{P}_{\mathrm{U}}$ be the solution space of (S-WTA) and (UB), and obj and obj_u be their respective optimal objective values, respectively. For target k, suppose that $z_{kl} = 1$ at one particular l. Then, constraints (17)–(19) of (UB) yield

$$a_{l+1} \geq A_k \geq a_l \text{ and } B_k \geq b_l.$$

Due to the minimum values that A_k and B_k may take are a_l and b_l, we have

$$A_k B_k \geq a_l b_l = u.$$

Therefore, if A_k and B_k satisfy constraints (17)–(20), constraint (6) is also satisfied. In other words, any feasible solution of (UB) is also feasible for (S-WTA). The analysis above means that $\mathbb{P} \supseteq \mathbb{P}_{\mathrm{U}}$. As formulations (S-WTA) and (UB) have the same objective and other constraints, it can be concluded that $obj \leq obj_u$. Thus, the optimal objective value of (UB) is a valid upper bound of the (S-WTA).

This work was supported in part by the National Natural Science Foundation of China under Grant 71971172, 71901176 and 72371200. (Corresponding author: Jianguang Feng.)

Yang Wang, Junpeng Wang, and Jianguang Feng are with the School of Management, Northwestern Polytechnical University, Xi'an 710072, China (e-mail: yangw@nwpu.edu.cn; wangjunpeng97@mail.nwpu.edu.cn; feng@nwpu.edu.cn). Jin-Kao Hao is with the Department of Computer Science, LERIA Lab, Université d'Angers, Angers 49045, France (e-mail: jin-kao.hao@univ-angers.fr).

II. Computational results

A. Robust analysis of ALNS

We give 10 independent runs for solving each instance and use Boxplots to summarize the experimental results. For each instance, we record the percentage difference of the objective value obtained in each run $\mathrm{Obj_r}$ from the average objective value among 10 runs $\mathrm{Obj_{avg}}$, which is calculated by $\mathrm{Difference\%} = (\mathrm{Obj_r} - \mathrm{Obj_{avg}})/\mathrm{Obj_{avg}} \times 100$. As shown in Figure 1, the percentage differences depend on the problem size and destruction probability. However, the maximum percentage difference does not exceed 5% for all the instances. This suggests that our ALNS algorithm presents good robustness in different runs.

B. Comparisons between ALNS and reference methods

We conduct an additional experiment using a running time of 300 seconds to compare the proposed ALNS algorithm with four reference methods. For instances where the optimal solution is known, the execution of each algorithm is terminated upon reaching the optimal values. We record the objective value found by each algorithm and the computational time required to attain this value. In addition, we record the objective percentage gap between ALNS and the other four algorithms, calculated by $(\mathrm{Obj^A} - \mathrm{Obj^{ALNS}})/\mathrm{Obj^{ALNS}} \times 100$, where $\mathrm{Obj^A}$ is the objective value of the comparison algorithms.

In Table I, a positive value of $gap\%$ in columns "UB model", "ALNS-H", "MRBCH" and "VLSN" means they are worse than ALNS. For a detailed analysis, we divide the 30 instances into small and large groups according to the instance size. The small group includes instances with no more than 200 sensors/weapons, while the large group includes the other instances.

TABLE I
COMPUTATIONAL RESULTS OF ALNS, UB MODEL, ALNS-H, MRBCH AND VLSN WITHIN 300s

Instances				ALNS (×10)		UB model			ALNS-H (×10)			MRBCH			VLSN		
S	W	T	u	Obj	Time(s)	Obj	Time(s)	Gap (%)	Obj	Time(s)	Gap (%)	Obj	Time(s)	Gap (%)	Obj	Time(s)	Gap (%)
Small instances																	
60	60	40	0.80	4111.2	23	**4110**	25	-0.03	4391.3	154	6.81	28537	0.01	594.13	5870	0.01	42.78
			0.85	5768.2	86	**5742**	244	-0.45	6633.0	177	14.99	-	0.02	-	12548	0.01	117.54
			0.90	9241.4	141	**8875**	197	-3.96	11787.6	129	27.55	-	0.01	-	25116	0.01	171.78
90	90	60	0.80	4043.3	90	**4040**	292	-0.08	4103.7	156	1.49	37945	0.06	838.47	5714	0.01	41.32
			0.85	5295.3	195	**5268**	288	-0.52	5647.4	187	6.65	42663	0.07	705.68	7788	0.01	47.07
			0.90	8819.4	175	**8330**	267	-5.55	11615.4	126	31.70	-	0.09	-	31196	0.03	253.72
120	120	80	0.80	4002.3	190	**4000**	161	-0.06	4184.0	159	4.54	53998	0.18	1249.17	5822	0.04	45.47
			0.85	5351.1	195	**5262**	184	-1.67	5579.4	227	4.27	58572	0.21	994.58	7399	0.04	38.27
			0.90	**8025.8**	183	8984	19	11.94	9354.9	181	16.56	-	0.27	-	17071	0.05	112.70
150	150	100	0.80	4337.0	197	**4284**	26	-1.22	4560.6	210	5.16	63499	0.44	1364.12	6105	0.1	40.77
			0.85	5325.4	178	**5259**	28	-1.25	5442.6	236	2.20	69955	0.49	1213.61	7596	0.12	42.64
			0.90	**8510.0**	213	9053	286	6.38	9737.8	245	14.43	-	0.61	-	19566	0.12	129.92
200	200	120	0.80	**3270.1**	168	3294	253	0.73	3421.7	204	4.64	76656	1.1	2244.15	5288	0.16	61.71
			0.85	4253.7	206	**4209**	53	-1.05	4499.4	223	5.78	85251	1.3	1904.16	6479	0.15	52.31
			0.90	**5882.1**	203	5956	80	1.26	6270.5	226	6.60	-	1.8	-	8710	0.3	48.08
Large instances																	
250	250	160	0.80	**3855.3**	216	3881	221	0.67	4145.6	253	7.53	98047	2.86	2443.17	6088	0.48	57.91
			0.85	**4961.5**	256	5037	233	1.52	5335.6	282	7.54	106639	3.46	2049.33	8362	0.41	68.54
			0.90	**7921.8**	241	15388	87	94.25	9067.6	260	14.46	-	4.63	-	22723	0.44	186.84
300	300	200	0.80	4225.4	244	**4194**	76	-0.74	4684.6	240	10.87	114567	6.27	2611.39	6270	1.1	48.39
			0.85	**5616.1**	283	5933	292	5.64	6122.6	245	9.02	133146	7.57	2270.79	8774	1.12	56.23
			0.90	**9097.5**	286	13437	156	47.70	10324.1	262	13.48	-	9.72	-	25203	1	177.03
350	350	250	0.80	4923.1	286	**4771**	243	-3.09	5387.3	275	9.43	145606	14.02	2857.61	7374	2.47	49.78
			0.85	**6801.1**	276	7324	173	7.69	7494.8	274	10.20	161508	16.11	2274.73	10463	1.91	53.84
			0.90	**12338.8**	291	27440	164	122.39	14696.9	278	19.11	-	19.57	-	30809	1.73	149.69
400	400	300	0.80	5575.5	283	**5575**	252	-0.01	6394.0	278	14.68	161917	27.69	2804.08	8142	3.76	46.03
			0.85	**7949.6**	274	14905	166	87.49	8864.4	256	11.51	184600	32.15	2222.13	12054	3.82	51.63
			0.90	**16002.2**	295	49659	160	210.33	20748.1	276	29.66	-	36.95	-	33706	4.04	110.63
500	500	360	0.80	5260.2	291	**5076**	176	-3.50	5832.4	275	10.88	199838	72.8	3699.06	7890	11.01	49.99
			0.85	**7093.9**	288	7864	150	10.86	8451.3	282	19.13	234056	85.56	3199.40	11312	9.97	59.46
			0.90	**13399.4**	293	28586	167	113.34	16470.0	277	22.92	-	87.25	-	33380	9.86	149.12

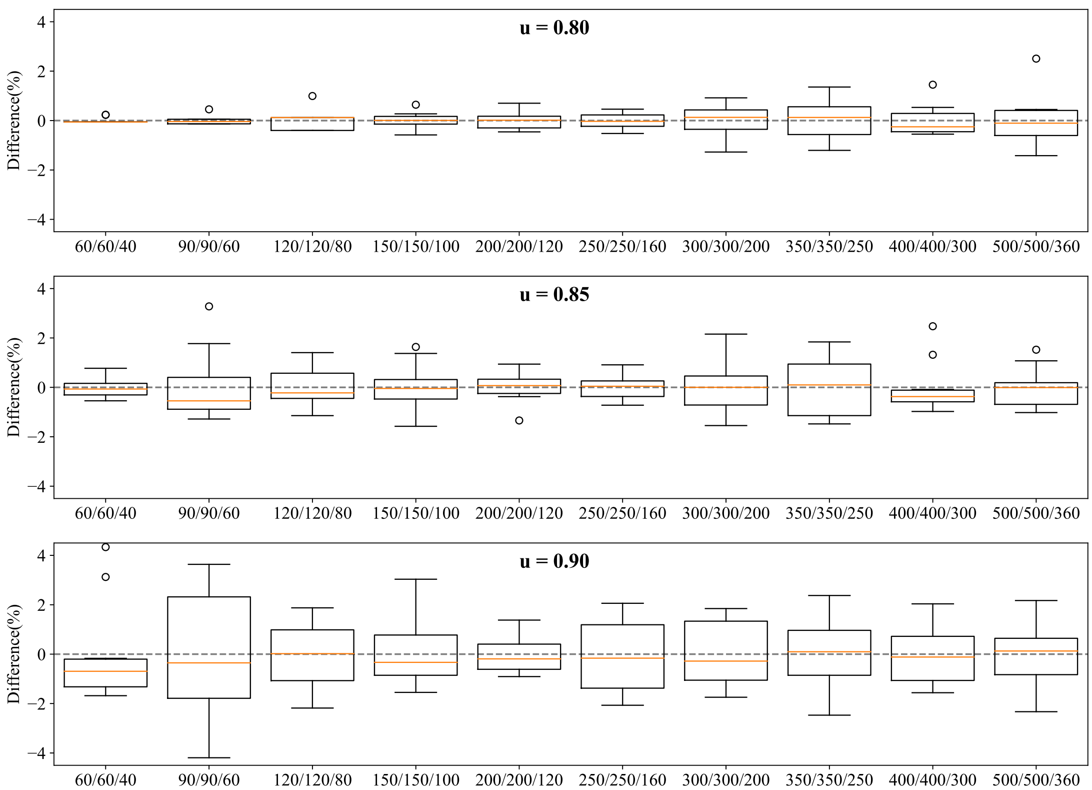


Fig. 1. The Boxplot for the repeated 10 runs of different instances.