Optimization of Weapon–Target Pairings Based on Kill Probabilities

Zbigniew R. Bogdanowicz, Antony Tolano, Ketula Patel, and Norman P. Coleman

***Abstract*—In this paper, we present a novel optimization algorithm for assigning weapons to targets based on desired kill probabilities. For the given weapons, targets, and desired kill probabilities, our optimization algorithm assigns weapons to targets that satisfy the desired kill probabilities and minimize the overkill. The minimization of overkill assures that any proper subset of the weapons assigned to a target results in a kill probability that is less than the desired kill probability on such a target. Computational results for up to 120 weapons and 120 targets indicate that the performance of this algorithm yields an average improvement in quality of solutions of 26.8% over the greedy algorithms, whereas execution times remained on the order of milliseconds.**

***Index Terms*—Auction algorithm, collaborative engagement, decision trees, discrete optimization, kill probability, parallel algorithms, scalable lethality, weapon-target assignment.**

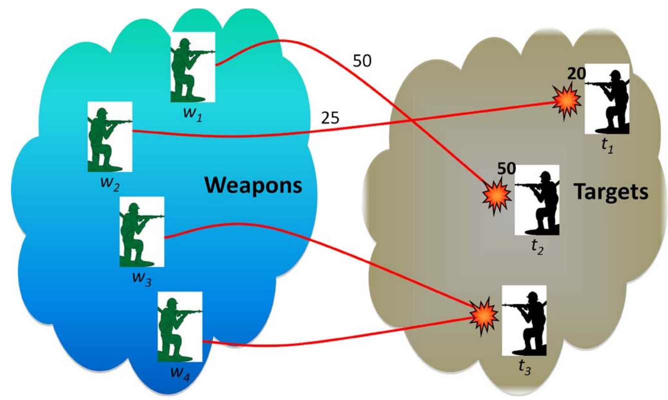


Fig. 1. Illustration of EWTP.

I. Introduction

In tomorrow's battlefields and combat scenarios, the collaborative engagement of many weapons (i.e., blue force) on many targets (i.e., red force) will play an ever increasing role. In fact, due to the advancements in the secure wireless communications, sensors, computational power, and robotics, the outcomes of the future battles will be decided predominantly by the intelligent pairing of the available weapons with the targets of interest to accomplish the desired lethality on such targets. Synchronized and intelligent collaborative engagement is and will be a key to winning the battles. Typically, commanders in the field determine the desired effects on targets in terms of desired percentage of damage [3], [10], probability of kill [33], etc. As the priority and strength of a target increase, commanders will assign more powerful desired effects to be applied on the target. Conversely, commanders will assign lesser desired effects as the probability of collateral damage (i.e., civilians, churches, schools, etc.) increases. As such, intelligent collaborative engagement of blue (friendly) forces on red (enemy) forces with a strong focus on scalable lethality will play a critical role in the future warfare.

Throughout this paper, we will use the term *weapon–target pairing* (*WTP*) referring to the weapon–target assignment (WTA) problem [1], [4], [7], [11]–[13], [16]–[20], [23]–[26], [28]–[32]. Consider a simple scenario for the effect-based WTP (EWTP) in Fig. 1, with four weapons w_1, w_2, w_3, and w_4, and three targets t_1, t_2, and t_3. Assume that weapons w_3 and w_4 must engage target t_3 in order to accomplish a desired predefined effect on t_3 (e.g., other weapons cannot achieve a desired effect on t_3 without weapons w_2 and w_3). Fig. 1 illustrates that many (i.e., two in this case) weapons can be assigned to a target to accomplish a desired effect. Suppose that w_1 has the effects equal to 30 and 50 (of some arbitrary units of effectiveness) on t_1 and t_2, respectively. Similarly, let w_2 have the effects equal to 25 and 40 on t_1 and t_2, respectively. Finally, let w_1 and w_2 together have the effects equal to 50 and 85 on t_1 and t_2, respectively. Note that the cumulative effect in this case is not additive. For the desired effects of at least 20 and 50 on t_1 and t_2, it is not wise to assign w_1 and w_2 to a single target. The desired effect would be achieved only for a single target with significant relative overkill. It is a natural solution to spread the assignment of our two weapons in this case to both targets: $w_2 \leftrightarrow t_1$, $w_1 \leftrightarrow t_2$, which is illustrated in Fig. 1. Therefore, we can satisfy desired effects on both targets rather than on a single one. Furthermore, assume the desired effects of 20 and 55 (on targets t_1, t_2 respectively) instead. In this case, assigning a single weapon to t_1 rather than both weapons to t_2 provides a couple of distinct advantages. This would allocate only a single weapon resource. In addition, the overkill of $w_2 \leftrightarrow t_1(5)$ is much lower than $w_1, w_2 \leftrightarrow t_2(20)$.

In this paper, we focus on effects that are "kill probabilities." Note that the kill probabilities are not additive. In particular, we introduce and study a new combinatorial algorithm derived from the auction algorithm [2], [5] to optimize the assignment of available weapons to identified targets for the given desired kill probabilities of such targets. As a result, our achieved kill

Manuscript received May 3, 2012; revised September 5, 2012 and November 14, 2012; accepted November 27, 2012. Date of publication December 21, 2012; date of current version November 18, 2013. This paper was recommended by Associate Editor H. Takagi.

The authors are with the Armament Research, Development and Engineering Center (ARDEC), Picatinny Arsenal, NJ 07806-5000 USA (e-mail: zbigniew.bogdanowicz.civ@mail.mil; antony.j.tolano.civ@mail.mil; ketula.patel.civ@mail.mil; norman.p.coleman4.civ@mail.mil).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2012.2231673

probabilities on the given targets satisfy desired kill probabilities on these targets (i.e., the achieved probabilities are greater or equal to the desired kill probabilities on the given targets) with minimal overkill.

Optimization of EWTPs that we describe in this paper is just one of many processes in the execution of the *kill chain* [15], [27] in military missions. Since there is a variety of potentially unpredictable targets that we can encounter, one of the essential components of the kill chain is target classification. It allows for a reduction of an unlimited variety of targets into finite set of the target types. Consequently, our effect-based weapon–target algorithm can be implemented in a manner that it acts on such finite set of the target types rather than on the targets themselves.

It is well known that the assignment of weapons to targets in order to accomplish the desired effects is NP-complete [22]. There are two simplified scenarios for which the exact algorithms have been investigated: 1) At most, a single weapon can be assigned to each target [8], [27]; and 2) all the weapons are identical [9]. In our case however, we need to find the best combination of heterogeneous weapons to be assigned to each target. This is clearly a more challenging task. Hence, we focus on the heuristic rather than the exact solution.

There are two types of the WTA problems widely covered in literature: static WTA [14], [30], and dynamic WTA [6], [21], [31], [32]. For static WTA, all inputs (i.e., weapons, targets, desired effects, engagement time, etc.) are given *a priori*. For dynamic WTA, a partial WTA is considered initially, followed by battle damage assessments, which might induce follow-on incremental WTAs. In this paper, we mainly focus on the static WTA problem, although our system architecture in Section III also supports the dynamic WTA problem. That is, by presenting several WTA solutions for different time instances, our system architecture allows the commander to choose/approve a preferred engagement solution at an appointed time.

The main focus of this paper is on the new optimization algorithm for the EWTP based on the kill probabilities. That is, for the given desired kill probabilities specified per target, we optimize the assignment of weapons to targets with the constraint that the desired kill probabilities are satisfied with minimum overkill. Our minimum overkill constraint implicitly supports the minimization of collateral damage. As such, we will incorporate a component that will assure that any collateral damage is minimized.

In [3], we derived an input for assigning weapons to targets based on the given effects of those weapons when applied to the targets and the desired effects on the targets. In [4] and [5], we covered assigning weapons and sensors to targets based on the given benefits of assigning these weapons/sensors to targets with the objective of maximizing the total assigned benefit. In fact, in the case of [4], we obtained the optimal assignment of weapons to targets but only allowed a single weapon to be assigned to a target. In this paper, we lifted this restriction, and we permit many weapons to be assigned to a target. Although, in [5], we also considered the assignment of many weapons to a target, in that paper, we did not specify the exact steps of the optimization algorithm, and we did not prove its convergence as we did here.

The significance of this paper is defined by the several original and innovative ideas that differ it from the previous published papers (such as [3]–[5]) in the following four core aspects. First, for the first time, we introduced integrated end-to-end assignment of weapons to targets based on the given weapons, targets, and desired kill probabilities. Second, we introduced a new pruning approach based on the upper bound for the number of weapons that can be assigned to a target. This makes our algorithm much more efficient in memory consumption and execution times, and makes it practical for use on lower powered hardware. Third, we designed and presented new parallel algorithms for EWTP that should further improve the performance and executions times of our currently implemented algorithms. Fourth, we included collateral damage considerations to arrive at the solution for effect-based assignment of weapons to targets.

The rest of this paper is organized as follows. In Section II, we describe the problem and formally state the objective of our optimization. In Section III, we present a system architecture that supports static and dynamic WTAs. Section IV represents the main body of this paper where we introduce and describe all our algorithms, and show that they converge. In Section V, we discuss a deconfliction consideration. Section VI provides and discusses the computational results based on the twelve test cases. Finally, in Section VII, we briefly summarize the main achievements of this paper.

II. Problem Description

The problem of optimized assignments of weapons to targets can be described by the given m weapons, n targets, and desired kill probabilities of these n targets. Let $K_1, K_2, \ldots, K_n$ be the given desired kill probabilities of n targets $t_1, t_2, \ldots, t_n$. Let $P_i(w_1), P_i(w_2), \ldots, P_i(w_m)$ be the given kill probabilities of target t_i by weapons $w_1, w_2, \ldots, w_m$, respectively. Let $P_i(q(i))$ denote a kill probability of target t_i by subset of weapons $S_i(q(i)) = \{w_{i_1}, w_{i_2}, \ldots, w_{i_{q(i)}}\}$. We claim that the following relation holds in the real world:

$$P_i(q(i)) \geq 1-(1-P_i(w_{i_1}))(1-P_i(w_{i_2}))\ldots\left(1-P_i(w_{i_{q(i)}})\right). \tag{1}$$

The reason for the above inequality is that, for any two weapons j and k, the probability of not killing t_i is actually less than $(1 - P_i(w_j))(1 - P_i(w_k))$. This is inferred from the possibility of hitting nonoverlapping regions of our target by both weapons, not killing t_i individually, and giving a possibility of exceeding a threshold of killing t_i collectively. Let $Q_i(q(i))$ be a lower bound for killing probability of t_i by weapons $w_{i_1}, w_{i_2}, \ldots, w_{i_{q(i)}}$ for given $P_i(w_1), P_i(w_2), \ldots, P_i(w_m)$. In particular, we assume $Q_i(q(i)) = 0$ if corresponding $S_i(q(i)) = \emptyset$. Otherwise, for $S_i(q(i)) \neq \emptyset$, we obtain the following:

$$Q_i\left(q(i)\right)=1-(1-P_i(w_{i_1}))(1-P_i(w_{i_2}))\ldots\left(1-P_i(w_{i_{q(i)}})\right). \tag{2}$$

Hence, we can formulate the optimization problem as follows:

$$\text{minimize} \sum_{i=1}^{n} \left|Q_i\left(q(i)\right) - K_i\right| \tag{3}$$

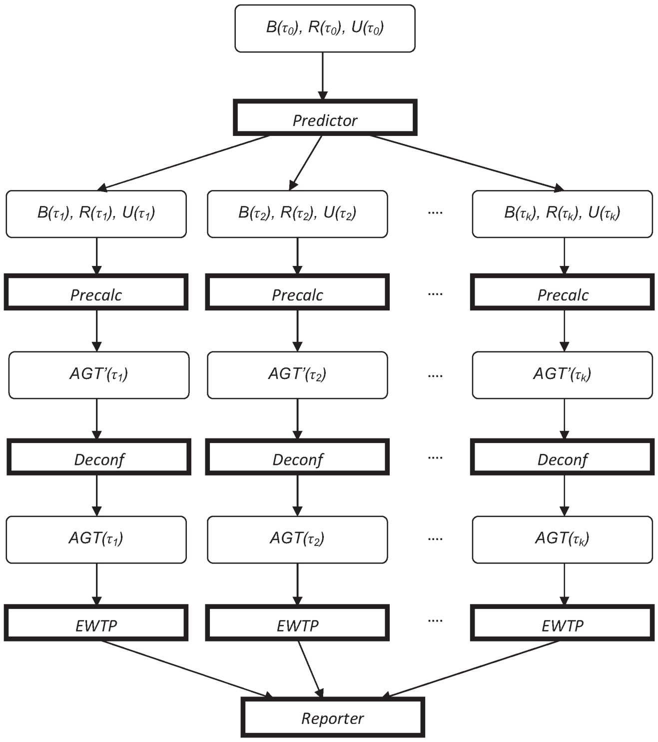


Fig. 2. Systems architecture supporting EWTP.

subject to

$$Q_i\left(q(i)\right) \geq K_i \text{ for } S_i\left(q(i)\right) \neq \emptyset \tag{4}$$

$$S_i\left(q(i)\right) \bigcap S_{i'}\left(q(i')\right) = \emptyset \text{ for } i \neq i'. \tag{5}$$

This is a combinatorial optimization problem because, for each target t_i in (3), it implies finding a subset of weapons $w_{i_1}, w_{i_2}, \ldots, w_{i_{q(i)}}$, which satisfies constraints (4) and (5). Constraint (4) assures that a desired effect is accomplished on target t_i, and constraint (5) prohibits a weapon to be assigned to multiple targets. The objective (3) of our optimization is to obtain the effects, which are the probabilities of kill, on targets $t_1, t_2, \ldots, t_n$ as close to the desired effects as possible. If all targets are assigned to some weapon combinations, then (3) minimizes the overkill attributed to the excess of probability of kill. If only one of two effects can be accomplished for some pair of targets with the same overkill value, then optimization (3) favors an effect with the higher value, which reflects real-world combat scenarios. That is, high-value targets usually have higher desired kill probabilities than the lower value targets.

III. Systems Architecture

The system architecture supporting our EWTP is illustrated in Fig. 2. We assume that the blue force, red force, and other friendly units (i.e., units that we do not want to destroy) are given. For each unit, the status of a unit, along with its location and orientation (i.e., pose of a unit), is known. Let $B(\tau_0)$, $R(\tau_0)$, $U(\tau_0)$ be the sets of blue, red, and friendly units, respectively, at time τ_0. Predictor(B, R, U) in Fig. 2 predicts, $B(\tau_i)$, $R(\tau_i)$, $U(\tau_i)$, which is the status and pose of the units at times $\tau_1, \tau_2, \ldots, \tau_k$ in the future based on $B(\tau_0)$, $R(\tau_0)$, $U(\tau_0)$. Precalc(B, R, U) represents the generation of attack guidance table (AGT) AGT$'(\tau_i)$. Deconf(B, R, U) takes into account deconflictions and modifies AGT$'(\tau_i)$, creating the AGT(τ_i) table that is subsequently used as the input for the EWTP optimizer. For each time τ_i, a series of the parallel executions, i.e.,

$$\text{Precalc} \rightarrow \text{Deconf} \rightarrow \text{EWTP}$$

is performed, which represents our optimized EWTP results. This optimization is the focus of this paper. In particular, in the following, Algorithms 1 and 2 represent Precalc, Algorithm 5 represents Deconf, and Algorithms 3 and 4 represent EWTP. Once the EWTP optimization is completed, the Reporter component visualizes and posts the results to the commander in the field for the approval of an appropriate further action.

IV. WTP Algorithm

Our EWTP optimization algorithm consists of two phases executed sequentially. In the first phase, an AGT is generated by executing *AGT_Gen* algorithm (with implicit consideration of deconfliction, which we describe in Section V). In the second phase, an optimized assignment of weapons to targets is generated (based on the AGT) by executing the *Min_Kill* algorithm.

Algorithm 1 Generate *AGT* (*AGT_Gen* algorithm)

Input: targets: $t_1, t_2, \ldots, t_n$,
weapons: $w_1, w_2, \ldots, w_m$,
upper bound for $q(1), q(2), \ldots, q(n)$: U,
effects: $P_1(w_1), P_1(w_2), \ldots, P_1(w_m)$,
$P_2(w_1), P_2(w_2), \ldots, P_2(w_m)$,
....
$P_n(w_1), P_n(w_2), \ldots, P_n(w_m)$,
desired effects: $K_1, K_2, \ldots, K_n$.

Output: *AGT*.

Step 1. Initialize *minimal set* matrix A by setting $a_{i,j} := \emptyset$ for each $a_{i,j} \in A$.
Step 2. Initialize benefit matrix B by setting $b_i^j := 0$ for each $b_i^j \in B$.
Step 3. Dispatch execution of depth-first search for the minimal sets to n cores.
Step 4. Wait for completion of depth-first searches from all n cores.
Step 5. Generate *AGT* by joining A and B, and STOP.

A. *AGT_Gen Algorithm*

We first focus on Phase I of the EWTP algorithm. For $w_r \in S_i(q(i))$, define $S_i(q(i), r) = S_i(q(i)) - \{w_r\}$, where $q(i) \geq r \geq 1$. Thus, for $S_i(q(i), r)$, there is a corresponding lower bound $Q_i(q(i), r)$. For given desired effect K_i, we define the minimal set as a subset of weapons $S_i(q(i))$ that satisfies $Q_i(q(i)) \geq K_i$ and $Q_i(q(i), r) < K_i$ for $q(i) \geq r \geq 1$. The purpose of the *AGT_Gen* algorithm is to generate a table

TABLE I
AGT

$S_i^1(q(i))$	b_i^1	$S_i^2(q(i))$	b_i^2	...	$S_i^n(q(i))$	b_i^n
$S_1^1(q(1))$	b_1^1	$S_1^2(q(1))$	b_1^2	...	$S_1^n(q(1))$	b_1^n
$S_2^1(q(2))$	b_2^1	$S_2^2(q(2))$	b_2^2	...	$S_2^n(q(2))$	b_2^n
$S_3^1(q(3))$	b_3^1	$S_3^2(q(3))$	b_3^2	...	$S_3^n(q(3))$	b_3^n
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
...	...	...	...	...	...	...
$S_r^1(q(r))$	b_r^1	$S_r^2(q(r))$	b_r^2	...	$S_r^n(q(r))$	b_r^n

(i.e., AGT) with minimal sets and corresponding benefits of assigning these minimal sets to the targets (see Table I). Let $S_i^j(q(i)) = \{w_{i_{(1,j)}}, w_{i_{(2,j)}}, \ldots, w_{i_{(q(i),j)}}\}$ be the jth minimal set defined for target t_i in the AGT, where $i_{(k,j)}$ denotes an index of the kth weapon in $S_i^j(q(i))$. Hence, $S_i^j(q(i))$ satisfies $Q_i^j(q(i)) \geq K_i$ and $Q_i^j(q(i), r) < K_i$ for $q(i) \geq r \geq 1$. Let b_i^j be a benefit (defined in Step 2 of Algorithm 2 and explained at the end of this section) of assigning the jth minimal set from the AGT to target t_i.

Unfortunately, based on the empirical results in [3], the number of the minimal sets grows exponentially and becomes impractical for tens of weapons. It is not the execution time but the size and the number of generated minimal sets that become a bottleneck. Memory requirements might become huge, and the weapons combinations processing by the follow-on weapon–target optimization algorithm might be very challenging. To overcome this serious issue, we introduce the global upper limit U on the number of weapons that can be assigned to any given target. As such, for each $q(i)$, we impose constraint $q(i) \leq U$, where U is user specified. We estimate that $U \leq 30$, which makes the outcome of the *AGT_Gen* algorithm practical. In fact, it was shown in [3] that the execution times for such a constraint runs on the order of nanoseconds, and the number of minimal sets runs up to hundreds. Furthermore, the memory requirements never exceeded 3 MB in our test executions of up to 120 weapons and 120 targets. This is clearly manageable.

AGT_Gen has been designed for multicore processors where the number of cores exceeds the maximum number of targets expected to be engaged for any given scenario. Each active core is associated with a unique target. From this point on, we assume that there are n cores (since inactive cores can be discarded), where n is the number of targets under consideration. We also assume that, for every weapon w_j, effect $P_i(w_j)$ (i.e., kill probability) on each target t_i is given. In addition, we assume that, for each target t_i, a desired effect K_i has been specified. *AGT_Gen* is based on the execution of depth-first search algorithm, where every vertex of a binary tree T (that needs to be built) corresponds to a weapon that can be assigned to a given target. Specifically, the root of T corresponds to some arbitrary weapon, and all 2^k vertices of distance k from the root correspond to a distinct/unique weapon. Therefore, the depth of T (i.e., the longest distance between the root and 2^k vertices for some $k \geq 1$) is equal to $m-1$, where m is the number of weapons. Having such a binary-tree structure, depth-first search is accomplished through the well-known depth-first traversal of T (i.e., either preorder, in-order, or postorder T traversal). Any vertex v of T is included in a current solution (i.e., in a currently generated minimal set) if, in a current traversal of T, either v is a leaf or a left child of v is chosen (see [3] for detailed description of depth-first search based on preorder traversal of T). The *AGT_Gen* algorithm executes as follows: Each depth-first search for the minimal sets is executed by core i dedicated to target i, as presented in Algorithm 2. The pruning is intertwined in the depth-first search based on $q(i) > U$, as implied by both steps. This algorithm translates to the generation of the ith row for matrices A and B, where each entry in row i of matrix A represents a unique minimal set of weapons assigned to target i, and each entry in row i of matrix B represents a corresponding benefit of assigning that minimal set of weapons from matrix A to target i.

Algorithm 2 Generate all minimal sets for target t_i by the ith core

Input: target: t_i,
weapons: $w_1, w_2, \ldots, w_m$,
upper bound for $q(i)$: U,
effects: $P_i(w_1), P_i(w_2), \ldots, P_i(w_m)$,
desired effect: K_i.
Output: matrices: A, B.

Step 1. Execute depth-first search for all the minimal sets with pruning based on $q(i) > U$ for target t_i.
Step 2. For every jth found, a minimal set that satisfies $|S_i^j(q(i))| \leq U$ do:
(i) save minimal set in A by setting

$$a_{i,j} := S_i^j(q(i));$$

(ii) save benefit in B by setting

$$b_i^j := 100 - Q_i^j(q(i)) + K_i;$$

Step 3. STOP.

Benefit b_i^j derived from $Q_i^j(q(i))$ is obtained from the depth-first search and from given K_i according to expression $b_i^j := 100 - Q_i^j(q(i)) + K_i$, where $100 \geq Q_i^j(q(i)) \geq K_i > 0$. Therefore, our derived benefits are real numbers that satisfy $100 \geq b_i^j > 0$. In particular, $b_i^j = 100$ if $Q_i^j(q(i)) = K_i$. This guarantees that, if we produce a greater total benefit in our assignment optimization algorithm *Min_Kill,* then it results in a smaller total overkill $\sum_{i=1}^n (Q_i(q(i)) - K_i)$ because of constraint (2). This in turn will allow minimization of overkill according to the objective (1) in *Min_Kill* (that we present in the next subsection) by maximizing a corresponding total benefit based on benefits b_i^j. The generation of benefits b_i^j can be accomplished even more efficiently by incorporating additional pruning into depth-first search [3] that does not eliminate any feasible minimal set satisfying $q(i) \leq U$. We have shown in [3] that such pruning can also save considerable time to generate the minimal sets.

The cumulative contribution of parallel execution of all n cores produces matrices A and B, with n rows corresponding to n targets and variable number of columns $m(i)$ per row i, which corresponds to the number of minimal sets generated for target i implied by row i.

B. *Min_Kill Algorithm*

The *Min_Kill* algorithm executes phase two of the EWTP optimization. Before presenting the *Min_Kill* algorithm, we introduce the *adjusted benefit* that will play a vital role in our algorithm. The *adjusted benefit* a_i^j of assigning minimal set S_i^j to target t_i is equal to benefit b_i^j reduced by the currently assigned scores for targets that have been assigned and share at least one weapon with S_i^j. The scores s_i are defined by Steps 1, 5, and 6 in *Min_Kill*. Initially, these scores are set to zero, and later, during iterations of *Min_Kill,* they increase and act as penalties associated with the reassignment of currently assigned targets. Their total value increases after each iteration, which typically maximizes the total benefit of complete assignment of n targets (this follows from the property of the auction algorithms [4]), which in turn minimizes the overkill of a complete assignment (as we explained in the earlier subsection). Let $s_1, s_2, \ldots, s_n$ be scores assigned to targets $t_1, t_2, \ldots, t_n$, respectively. Then, we have

$$a_i^j = b_i^j - \sum_{t_k \neq t_i} \left(s_k | \exists w, w \leftrightarrow t_k \text{ and } w \in S_i^j \right) \tag{6}$$

where $w \leftrightarrow t_k$ denotes that weapon w is currently assigned to target t_k. Note that, in our notation, a_i^j denotes an adjusted benefit, whereas $a_{i,j}$ denotes a minimal set.

Algorithm 3 Serial optimizer (*Min_Kill* algorithm)

Input: *AGT*.
Output: Assignment of minimal sets to targets.

Step 1. Initialize target scores $s_1 := s_2 := \ldots := s_n := 0$.
Step 2. If there exists unassigned target t_i with the associated minimal set of positive adjusted benefit, then execute Steps 3–9. Otherwise, STOP.
Step 3. Find j that maximizes adjusted benefit a_i^j for t_i.
Step 4. Find $j' \neq j$ that maximizes adjusted benefit $a_i^{j'}$ for t_i.
Step 5. For every target t_k (where $t_k \neq t_i$)) with currently assigned weapon $w_{k'} \in S_i^j(q(i))$ do:
(i) $s_i := s_i + s_k$;
(ii) reset score $s_k := 0$;
(iii) unassign assigned minimal set S_k;
(iv) for every weapon $w_{k''}$ currently assigned to t_k unassign $w_{k''}$.
Step 6. If j' exists, then calculate score for target t_i as follows:

$$s_i := s_i + b_i^j - max(0, a_i^{j'}) + \epsilon.$$

Else, calculate score for target t_i as follows:

$$s_i := s_i + b_i^j + \epsilon.$$

Step 7. Assign minimal set $S_i^j(q(i))$ to target t_i;

$$S_i^j(q(i)) \leftrightarrow t_i.$$

Step 8. For every weapon $w_{k'} \in S_i^j(q(i))$ assign $w_{k'}$ to target t_i; $w_{k'} \leftrightarrow t_i$.
Step 9. Go to Step 2.

Min_Kill optimization executes in the nine steps illustrated in Algorithm 3. This algorithm has been derived from the auction algorithm [2], [5]. At each iteration (steps 2–9), *Min_Kill* assigns minimal set $S_i^j(q(i))$ to one of the unassigned targets t_i. Depending on the situation, zero, one, or more targets can be unassigned as a result of $S_i^j(q(i)) \leftrightarrow t_i$, which is performed by Step 5.

C. *Convergence of the Min_Kill Algorithm*

Define total score $S = \sum_{i=1}^{n} s_i$, where n denotes the number of identified targets. Our next result will be based on examining S in *Min_Kill* after each iteration.

Theorem 1: Algorithm 3 converges to a feasible optimized assignment in a finite number of steps.

Proof: Let $b_{\max}$ be the largest b_i^j in the AGT. Clearly, our total score S cannot exceed $n \times (b_{\max} + \epsilon)$ based on the above definition of S. Initially, $S = 0$. After first iteration of *Min_Kill*, $S = s_i > 0$. Suppose that, after iteration j, $S = Q_j$ for some positive number Q_j. If *Min_Kill* does not terminate in Step 2 during $j + 1$ iteration, then S will increase by at least ϵ due to Steps 5–6. Therefore, after iteration $j + 1$, a total score $S > Q_j$. Hence, by induction and $S \leq n \times (b_{\max} + \epsilon)$, *Min_Kill* terminates. ■

Consider now the worst execution time $T_{\max}$ of *Min_Kill*. Let $q_{\max}$ denote the largest number of minimal sets for any given target. Therefore, according to Steps 3–4, there are $O(q_{\max})$ operations to find and process the best and second best assignments for an unassigned target per iteration. In addition, Step 5 requires $O(n \times m)$ operations to unassign previously assigned targets based on reused weapons, where m denotes the total number of weapons. Since $S \leq n \times (b_{\max} + \epsilon)$, and at each iteration, S increases by at least ϵ, then

$$T_{\max} = O\left(\frac{(q_{\max} + nm) n b_{\max}}{\epsilon}\right). \tag{7}$$

Note that $T_{\max}$ of *Min_Kill* is quite attractive since, in our case, $b_{\max}$ is related to overkill (recall Step 2 in Algorithm 2). Thus, $b_{\max} \leq 100$, and the worst execution time for *Min_Kill* simplifies to

$$T_{\max} = O\left(\frac{n q_{\max} + n^2 m}{\epsilon}\right). \tag{8}$$

TABLE II
EXAMPLE OF AGT

$S_i^1(q(i),1)$	b_i^1	$S_i^2(q(i),2)$	b_i^2
$\{w_2,w_3\}$	71	$\{w_1,w_4\}$	31
$\{w_2,w_5\}$	80	$\{w_3,w_6\}$	50

D. Example of WTP Optimization

Consider a simple example of the AGT with two rows and two pairs of columns for each row, as illustrated in Table II. Initially, benefits b_i^j from the *AGT* are equal to adjusted benefits a_i^j. Assume that an implementation of *Min_Kill* scans rows of the AGT (corresponding to targets $t_1$andt_2) from top to bottom in a round-robin fashion. Based on these assumptions, the iterations of *Min_Kill* execute as follows:

Iteration 1: Target t_1 is processed. Since $a_1^1 = (71-0) > (31-0) = a_1^2$, then score s_1 is calculated in Step 6 according to $s_1 := 0 + (71-0) - (31-0) + \epsilon = 40 + \epsilon$. In Steps 7–8, minimal set $S_1^1(2)$ and its weapons $w_2$andw_3 are assigned to target t_1, i.e., $\{w_2,w_3\} \leftrightarrow t_1$, $w_2 \leftrightarrow t_1$, $w_3 \leftrightarrow t_1$. The total score S after this iteration is $S = s_1 = 40 + \epsilon$.

Iteration 2: Target t_2 is processed. Now, the following relation for adjusted benefits holds: $a_2^1 = (80 - (40+\epsilon)) > (50 - (40+\epsilon)) = a_2^2$. Therefore, in Step 5, we obtain $s_2 := s_2 + s_1 = 0 + (40+\epsilon) = 40 + \epsilon$, and in Step 6, we update $s_2 := (40+\epsilon) + (80 - (40+\epsilon)) - (50 - (40+\epsilon)) + \epsilon = 70 + 2\epsilon$. In Steps 7–8, minimal set $S_2^1(2)$ and its weapons w_2 and w_5 are assigned to target t_2, i.e., $\{w_2,w_5\} \leftrightarrow t_2$, $w_2 \leftrightarrow t_2$, $w_5 \leftrightarrow t_2$. The total score S after this iteration is $S = s_2 = 70 + 2\epsilon$.

Iteration 3: Target t_1 is processed according to a round-robin scheme. Now, $a_1^1 = (71 - (70 - 2\epsilon)) < (31 - 0) = a_1^2$ holds. Step 5 is skipped, and in Step 6, we obtain $s_1 := 0 + (31-0) - (71 - (70 + 2\epsilon)) + \epsilon = 30 + 3\epsilon$. In Steps 7–8, minimal set $S_1^2(2)$ and its weapons w_1 and w_4 are assigned to target t_1, i.e., $\{w_1,w_4\} \leftrightarrow t_1$, $w_1 \leftrightarrow t_1$, $w_4 \leftrightarrow t_1$. The total score S after this iteration is $S = s_1 + s_2 = 100 + 5\epsilon$, and *Min_Kill* stops.

E. Parallel Optimizer

In order to present the parallel version of the *Min_Kill* algorithm (i.e., Algorithm 4), which we call *Min_Kill_Par*, we introduce one more definition. If there exists a positive adjusted benefit for target t_i, then *benefit gain* G_i is the difference between the best adjusted benefit and the second best adjusted benefit induced by the assignment of two minimal sets to t_i. Let $a_i(1) = \max_j(a_i^j)$ and $a_i(2) = \max_{j'}(a_i^{j'})|_{j' \neq j}$. Therefore,

$$G_i = a_i(1) - \max\left(0, a_i(2)\right).$$

The main reason of introducing *Min_Kill_Par* is to improve the quality of solution without sacrificing the execution time. The idea is that assigning a minimal set that maximizes either G_i or a_i (based on *a priori* selected criterion) at each iteration i tends to result in a better quality of solution on the average. This conjecture, however, needs to be verified empirically. If it holds, then *Min_Kill_Par* should also significantly gain an execution time advantage over *Min_Kill*, assuming that *Min_Kill* would need to achieve the same quality of solution.

Algorithm 4 Parallel optimizer (*Min_Kill_Par* algorithm)

Input: *AGT*.
Output: Assignment of minimal sets to targets.

Step 1. Initialize target scores $s_1 := s_2 := \ldots := s_n := 0$.
Step 2. If there exists unassigned target t_i with associated minimal set of positive adjusted benefit then execute Steps 3–11. Otherwise, STOP.
Step 3. For every unassigned t_r dispatch to core r, search for $a_r(1), a_r(2)$.
Step 4. Wait until all dispatched tasks to the cores are completed.
Step 5. Pick unassigned target t_i based on *a priori* selection criterion:
  (i) $\max_r(a_r(1))$; or
  (ii) $\max_r(G_r)$.
Step 6. Identify j that maximizes adjusted benefit a_i^j for t_i.
Step 7. For every target t_k with currently assigned weapon $w_{k'} \in S_i^j(q(i))$, do:
  (i) $s_i := s_i + s_k$;
  (ii) reset score $s_k := 0$;
  (iii) unassign assigned minimal set S_k;
  (iv) for every weapon $w_{k''}$ currently assigned to t_k unassign $w_{k''}$.
Step 8. If j' exists then calculate score for target t_i as follows:

$$s_i := s_i + b_i^j - max(0, a_i^{j'}) + \epsilon.$$

Else, calculate score for target t_i as follows:

$$s_i := s_i + b_i^j + \epsilon.$$

Step 9. Assign minimal set $S_i^j(q(i))$ to target t_i.
Step 10. For every weapon $w_{k'} \in S_i^j(q(i))$ assign $w_{k'}$ to target t_i.
Step 11. Go to Step 2.

F. Example for Parallel Optimizer

Consider again the example from Section IV-D (i.e., Table II), this time applied to the parallel optimizer. We show that for both parallel strategies of picking the best unassigned target in Step 5 of *Min_Kill_Par* (i.e., largest adjusted benefit versus largest benefit gain), our algorithm needs reassignments. For strategy (ii) $\max_r(G_r)$ in Step 5, the iterations execute exactly the same as in the serial implementation; therefore, they are exactly the same as in the example in Section IV-D, where reassignment in Iteration 2 was taking place. Hence, we assume strategy (i) $max_r(a_r(1))$ in Step 5. In this case, the iterations of *Min_Kill_Par* execute as follows:

Iteration 1: Target t_2 is processed, and $a_2^1 = (80-0) > (50-0) = a_2^2$. Therefore, score s_2 is calculated in Step 8 according to $s_2 := 0 + (80 - 0) - (50 - 0) + \epsilon = 30 + \epsilon$. In Steps 9–10, minimal set $S_2^1(2)$ and its weapons w_2, w_5 are assigned to target t_2, i.e., $\{w_2, w_5\} \leftrightarrow t_2$, $w_2 \leftrightarrow t_2$, $w_5 \leftrightarrow t_2$. The total score S after this iteration is $S = s_2 = 30 + \epsilon$.

Iteration 2: Target t_1 is processed, and $a_1^1 = (71 - (30 + \epsilon)) > (31 - 0) = a_1^2$. The score s_1 is calculated in Step 6 according to $s_1 := 0 + s_2 = 30 + \epsilon$. In Steps 9–10, minimal set $S_1^1(2)$ and its weapons w_2, w_3 are assigned to target t_1, i.e., $\{w_2, w_3\} \leftrightarrow t_1, w_2 \leftrightarrow t_1, w_3 \leftrightarrow t_1$. The total score S after this iteration is $S = s_1 = 40 + \epsilon$.

Iteration 3: Target t_2 is processed. Now, the following relation for adjusted benefits holds: $a_2^1 = (80 - (40 + \epsilon)) > (50 - (40 + \epsilon)) = a_2^2$. Therefore, in Step 7, we obtain $s_2 := s_2 + s_1 = 0 + (40 + \epsilon) = 40 + \epsilon$, and in Step 8, we update $s_2 := (40 + \epsilon) + (80 - (40 + \epsilon)) - (50 - (40 + \epsilon)) + \epsilon = 70 + 2\epsilon$. In Steps 9–10, minimal set $S_2^1(2)$ and its weapons w_2, w_5 are assigned to target t_2, i.e., $\{w_2, w_5\} \leftrightarrow t_2$, $w_2 \leftrightarrow t_2$, $w_5 \leftrightarrow t_2$. The total score S after this iteration is $S = s_2 = 70 + 2\epsilon$.

Iteration 4: Target t_1 is processed. Now, $a_1^1 = (71 - (70 - 2\epsilon)) < (31 - 0) = a_1^2$ holds. Step 5 is skipped, and in Step 6, we obtain $s_1 := 0 + (31 - 0) - (71 - (70 + 2\epsilon)) + \epsilon = 30 + 3\epsilon$. In Steps 9–10, minimal set $S_1^2(2)$ and its weapons w_1, w_4 are assigned to target t_1, i.e., $\{w_1, w_4\} \leftrightarrow t_1, w_1 \leftrightarrow t_1, w_4 \leftrightarrow t_1$. The total score S after this iteration is $S = s_1 + s_2 = 100 + 5\epsilon$, and *Min_Kill* stops.

V. Deconfliction Considerations

One of the key considerations before engaging targets on the battlefield is making sure that friendly units are not harmed or killed. In addition, a serious consideration has to be given to avoiding the destruction of certain structures such as places of worship, schools, etc. Collectively, these considerations are called deconflictions. We incorporated deconflictions with two flavors into EWTP, giving a user two options as follows. In Option 1, any subset of weapons in the *AGT* does not contain a weapon that would violate a deconfliction. This option assures that the proposed weapon–target solution generated by our optimizers *Min_Kill* and *Min_Kill_Par* will not violate a deconfliction. In Option 2, we generate an additional matrix of flags at the time when algorithm *AGT_Gen* generates the minimal sets. That is, to every minimal set S_i^j in *AGT_Table*, there corresponds a deconfliction flag d_i^j that is equal to zero if no weapon in S_i^j causes a deconfliction problem. Otherwise, d_i^j flag is set to 1 (see Algorithm 5). In this option, the weapon–target solution is presented to the commander in charge along with the alerts indicating minimal sets S_i^j assigned to targets t_i with the corresponding deconfliction flags $d_i^j = 1$.

For the given weapon w and target t, let $f(w, t)$ define a lethality area around target t. In our implementation, $f(w, t)$ is defined by radius R of the sphere with a center at the location of target t. Let $g(u, t)$ be a distance of a friendly unit from target t in a straight line. Based on our definitions, every unit u with $g(u, t) < f(w, t) = R$ will be destroyed if weapon w would engage and hit target t. Hence, the algorithm that generates the deconfliction flags d_i^j for the given minimal sets S_i^j in *AGT_Gen* looks as follows.

Algorithm 5 Set deconfliction flag d_i^j for minimal set S_i^j

Input: minimal set: S_i^j,
lethality: $f(w_1, t_i), f(w_2, t_i), \ldots, f(w_m, t_i)$,
friendly units: $u_1, u_2, \ldots, u_x$.
Output: Deconfliction flag d_i^j in matrix D.

Step 1. Set $R_i :=$ LARGE.
Step 2. For every friendly unit u_k do:
calculate $R_i := min(R_i, g(u_k, t_i))$.
Step 3. Set $d_i^j := 0$.
Step 4. For every weapon w_k in minimal set S_i^j do:
if $f(w_k, t_i) > R_i$ then

$$d_i^j := 1.$$

Step 5. STOP.

In Algorithm 5, the lethality of weapons in respect to a specific type of target is given. This is typically available from a particular weapon-related effect-based database (e.g., JMEM).

VI. Computational Results

We implemented *AGT_Gen* and *Min_Kill* algorithms in a RedHat Enterprise Linux 6.1 environment on a PC with an Intel(R) E8600 at 3.33-GHz CPU. We executed *AGT_Gen* for up to 120 weapons. In addition, we executed the optimization *Min_Kill* algorithm for twelve test regions defined by the minimum/maximum number of targets n and the minimum/ maximum number of sets per target q (i.e., first four columns in Table III). We varied n and q for up to 120 targets and minimal sets per target, respectively. For each region, we executed at least three test cases for the total of 141 test cases.

The execution of *AGT_Gen* per target was on the order of nanoseconds (see [3]), which makes it practical for military applications requiring near instantaneous feedback. We also observed that the execution times of *AGT_Gen* generally increased when a desired effect increased, but they were negligible in comparison with the execution times for *Min_Kill*. In order to enhance the visualization of execution time versus input size for *Min_Kill*, we plotted the values of time in milliseconds on the logarithmic scale shown in Fig. 3.

To evaluate the quality of the solution, we implemented a simple greedy algorithm to assign weapons to targets, and we retrieved an old greedy algorithm from our system of record that does the same. Then, we measured the percentages of improvement of *Min_Kill* with respect to our greedy algorithms in Table III for the same inputs and made sure that all targets were assigned in all algorithms for a fair comparison. Based on Table III, *Min_Kill* produced on the average 26.8% improvement over our greedy algorithms, and the average execution time of optimization was 55.8 ms. Note that the quality of

TABLE III
COMPUTATIONAL RESULTS FOR *Min_Kill* ALGORITHM

Min n	Max n	Min q	Max q	Min Exec Time [ms]	Max Exec Time [ms]	Best Imp. [%]	Worst Imp. [%]
1	10	2	10	1	1	47	0
11	20	11	20	3	4	64	0
21	30	21	30	5	8	65	22
31	40	31	40	9	15	74	23
41	50	41	50	16	22	63	12
51	60	51	60	27	36	64	4
61	70	61	70	39	51	75	11
71	80	71	80	55	66	48	15
81	90	81	90	72	91	54	12
91	100	91	100	96	116	44	2
101	110	101	110	123	150	43	14
111	120	111	120	155	177	54	14

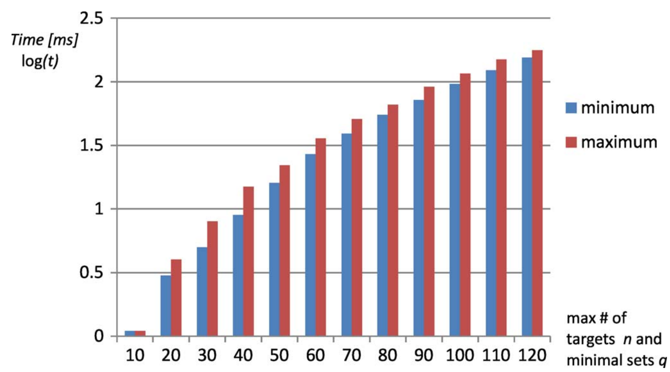


Fig. 3. Time versus input size.

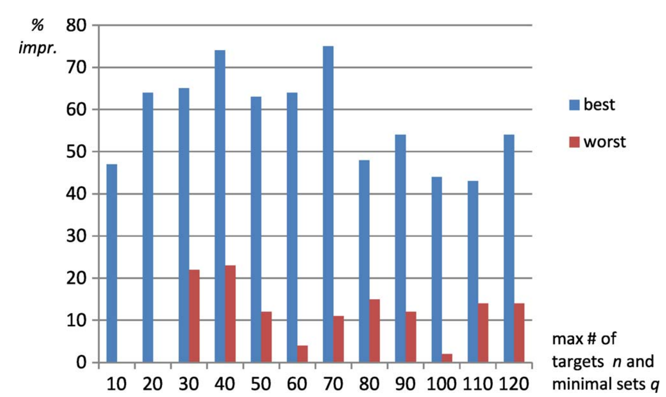


Fig. 4. Improvement percentage versus input size.

solution indicating a single-digit improvement for the worst cases in Fig. 4 (and corresponding to Table III) is actually better than a straightforward intuitive interpretation might suggest. This is because our percentage of improvement pertains to the probability of kill. For example, a weapon that required 95% of kill probability of a given target type might need to be much more sophisticated and, hence, much more expensive than a weapon requiring a 90% kill probability of the same target type. Therefore, a single-digit percentage improvement in the probability of kill might significantly reduce the cost of targets engagement; hence, it can be economically very attractive.

Our implementation of *Min_Kill* produced all solutions for up to 120 targets and 120 minimal sets on the order of milliseconds, and resulted in the optimized solution of up to a 75% improvement over our greedy solutions. We did not observe any obvious dependence between the improvement percentage of the solution and the input size (see Fig. 4).

The results for *Min_Kill* were obtained by independently generating random *AGT*s in the desired ranges for minimal sets. However, we note that, without any constraints even for tens of weapons, the number of minimal sets generated can greatly exceed the regions that we considered in Table III, as documented in [3]. We resolved this issue by imposing an upper bound on the number of weapons that can be assigned to a target. This restriction not only alleviates this problem but it also tends to produce realistic and desirable engagement solutions (i.e., solutions with not too many weapons assigned to a target). We verified that the memory (RAM) requirement was between 1 MB and 3 MB for up to 120 weapons and 120 targets.

Finally, our parallel algorithm *Min_Kill_Par* should more quickly converge to a comparable quality of solution. Hence, further performance evaluation in terms of execution time and the quality of solution for *Min_Kill_Par* is anticipated once it is implemented.

VII. CONCLUSION

We introduced a new optimization framework for EWTP that takes into consideration the kill probabilities. The objective of this optimization framework is to assign weapons to targets in such a way that the given desired kill probabilities on targets are satisfied with minimum overkill. Our framework consists of two phases, which leverage parallel and distributed processing. In Phase I, Algorithms 1 and 2 based on depth-first search efficiently generate minimal sets by employing intertwined pruning within the decision tree. As a side effect of such a pruning, the minimal sets generated are practical for optimization by imposing an upper bound on the number of weapons that can be assigned to any given target. In Phase II, Algorithm 3 (or corresponding parallel Algorithm 4) derived from the auction algorithm (but not equivalent to it) takes as input the *AGT* generated in Phase I and optimizes the assignment of weapons to targets with the objective to minimize overkill and with the constraint that the kill probabilities are satisfied.

The computational results indicate that the generation of the AGT (i.e., Algorithms 1 and 2) executes in nanoseconds, which is negligible in comparison with the optimization times of our serial optimizer (i.e., Algorithm 3). The average execution time for Algorithm 3 was 55.8 ms for up to 120 weapons and 120 targets, and its average improvement over the greedy algorithms was 26.8%. Employing our parallel optimizer, Algorithm 4 should result in additional execution time saving and an improvement of the quality of optimization. Consequently, the solution based on our parallel algorithms (i.e., Algorithms 1 and 2, and 4) should be quite attractive for military missions that involve hundreds of weapons and/or targets.

Because the percentage of improvement pertains to the probability of kill, our worst improvements based on Algorithm 3 (i.e., serial optimizer) are actually better than a straightforward intuitive interpretation might suggest. In particular, the improvements are economically attractive when one considers the cost effectiveness of small percentage gains when applied to weapons and munition costs required to engage high-kill-probability targets. In addition, our parallel algorithm *Min_Kill_Par* (i.e., Algorithm 4) should quickly converge and further improve the quality of solution. We plan to execute a thorough follow-on performance evaluation of the *Min_Kill_Par* algorithm in terms of execution time and quality of solution.

Acknowledgment

We would like to thank all referees for their valuable comments, which resulted in significant improvements to quality, robustness, and clarity of this paper.

References

[1] R. K. Ahuja, A. Kumar, K. C. Jha, and J. B. Orlin, "Exact and heuristic algorithms for the weapon–target assignment problem," *Oper. Res.*, vol. 55, no. 6, pp. 1136–1146, Nov./Dec. 2007.

[2] D. P. Bertsekas, "The auction algorithm for assignment and other network flow problems," *Interfaces*, vol. 20, no. 4, pp. 133–149, Jul./Aug. 1990.

[3] Z. R. Bogdanowicz, "Advanced input generating algorithm for effect-based weapon–target pairing optimization," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 42, no. 1, pp. 276–280, Jan. 2012.

[4] Z. R. Bogdanowicz, "A new efficient algorithm for optimal assignment of smart weapons to targets," *Comput. Math. Appl.*, vol. 58, no. 10, pp. 1965–1969, Nov. 2009.

[5] Z. R. Bogdanowicz and N. P. Coleman, "Optimization of sensor/weapon–target pairings based on auction algorithm," *WSEAS Trans. Math.*, vol. 6, no. 6, pp. 730–735, 2007.

[6] H. Cai, J. Liu, Y. Chen, and H. Wang, "Survey of the research on dynamic weapon–target assignment problem," *J. Syst. Eng. Electron.*, vol. 17, no. 3, pp. 559–565, Sep. 2006.

[7] E. Cetin and S. T. Esen, "A weapon–target assignment approach to media allocation," *Appl. Math. Comput.*, vol. 175, no. 2, pp. 1266–1275, Apr. 2006.

[8] S. Chang, R. James, and J. Shaw, "Assignment algorithm for kinetic energy weapons in boost defense," in *Proc. IEEE 26th Conf. Decision Control*, Los Angeles, CA, 1987, pp. 1678–1683.

[9] G. DenBroader, R. Ellison, and L. Emerling, "On optimum target assignments," *Oper. Res.*, vol. 7, no. 3, pp. 322–326, May/Jun. 1958.

[10] K. Deep and M. Pant, "Maximization of expected target damage value," *Defense Sci. J.*, vol. 55, no. 2, pp. 133–139, Apr. 2005.

[11] E. Erdem and N. E. Ozdemirel, "An evolutionary approach for the target allocation problem," *J. Oper. Res. Soc.*, vol. 54, no. 9, pp. 958–969, Sep. 2003.

[12] S. Han, Y. Liu, X. Yang, and J. Liu, "WTA problem in the warship fleet," *Fire Control Command Control*, vol. 34, no. 2, pp. 32–35, 2009.

[13] C. Huaiping and C. Yingwu, "The development of the research on weapon-target assignment problem," *Fire Control Command Control*, vol. 31, no. 12, pp. 11–15, 2006.

[14] F. Johansson and G. Falkman, "A suite of metaheuristic algorithms for static weapon–target allocation," in *Proc. Int. Conf. Genet. Evol. Method*, Las Vegas, NV, 2010, pp. 132–138.

[15] O. Karasakal, N. E. Ozdemirel, and L. Kandiller, "Anti-ship missile defense for a naval task group," *Naval Res. Logist.*, vol. 58, no. 3, pp. 305–322, 2003.

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

[17] M. Z. Lee, "Constraint weapon–target assignment: Enhanced very large scale neighborhood search algorithm," *IEEE Trans. Syst., Man, Cybern. A, Syst. Humans*, vol. 40, no. 1, pp. 198–204, Jan. 2010.

[18] Z. J. Lee, S. F. Su, and C. Y. Lee, "Efficiently solving general weapon–target assignment problem by genetic algorithms with greedy eugenics," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 33, no. 1, pp. 113–121, Feb. 2003.

[19] Z. J. Lee, S. F. Su, and C. Y. Lee, "A genetic algorithm with domain knowledge for weapon–target assignment problems," *J. Chin. Inst. Eng.*, vol. 25, no. 3, pp. 287–295, 2002.

[20] Z. J. Lee, C. Y. Lee, and S. F. Su, "An immunity-based ant colony optimization algorithm for solving weapon–target assignment problem," *Appl. Soft Comput. J.*, vol. 2, no. 1, pp. 39–47, Aug. 2002.

[21] J. Li, R. Cong, and J. Xiong, "Dynamic WTA optimization model of air defense operation of warships' formation," *J. Syst. Eng. Electron.*, vol. 17, no. 1, pp. 126–131, Mar. 2007.

[22] S. Lloyd and H. Witsenhausen, "Weapon allocation is NP-complete," in *Proc. Summer Conf. Simul.*, Reno, NV, 1986, pp. 1054–1058.

[23] A. M. Madni and M. Andrecut, "Efficient heuristic approaches to the weapon–target assignment problem," *J. Aerosp. Comput. Inf. Commun.*, vol. 6, no. 6, pp. 405–414, 2009.

[24] A. Malhotra and R. K. Jain, "Genetic algorithm for optimal weapon allocation in multilayer defense scenario," *Defense Sci. J.*, vol. 51, no. 3, pp. 285–293, 2001.

[25] A. S. Manne, "A target assignment problem," *Oper. Res.*, vol. 6, no. 3, pp. 346–351, May/Jun. 1958.

[26] H. I. Mekawey, M. S. Abd El-Wahab, and M. Hashem, "Novel goal-based weapon–target assignment doctrine," *J. Aerosp. Comput. Inf. Commun.*, vol. 6, no. 1, pp. 2–29, 2009.

[27] D. Orlin, "Optimal weapons allocation against layered defenses," *Naval Res. Logist.*, vol. 34, no. 6, pp. 605–616, 2007.

[28] R. K. Ahuja, A. Kumar, K. C. Jha, and J. B. Orlin, "Exact and heuristic algorithms for the weapon–target assignment problem," *Oper. Res.*, vol. 55, no. 6, pp. 1136–1146, Nov./Dec. 2006.

[29] J. M. Rosenberg, H. S. Hwang, R. P. Pallerla, A. Yucel, R. L. Wilson, and E. G. Brungardt, "The generalized weapon–target assignment problem," in *Proc. 10th Int. Command Control Res. Tech. Symp.*, McLean, VA, 2005, pp. 1–12.

[30] E. Wacholder, "A neural-based optimization algorithm for the static weapon–target assignment problem," *INFORMS J. Comput.*, vol. 1, no. 4, pp. 232–246, 1989.

[31] B. Xin, J. Chen, J. Zhang, L. Dou, and Z. Peng, "Efficient decision making for dynamic weapon–target assignment by virtual permutation and Tabu search heuristics," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 40, no. 6, pp. 649–662, Nov. 2010.

[32] B. Xin, J. Chen, Z. H. Peng, L. Dou, and J. Zhang, "An efficient rule-based constructive heuristic to solve dynamic weapon–target assignment problem," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 41, no. 3, pp. 598–606, May 2011.

[33] W. Yanxia, Q. Longjun, G. Zhi, and M. Lifeng, "Weapon target assignment problem satisfying expected damage probabilities based on ant colony algorithm," *J. Syst. Eng. Electron.*, vol. 19, no. 5, pp. 939–944, Oct. 2008.

Zbigniew R. Bogdanowicz received the M.S. degree in electronics engineering from the Wroclaw University of Technology, Wroclaw, Poland, and the Ph.D. degree in electrical engineering from the Stevens Institute of Technology, Hoboken, NJ.

For many years, he was a Distinguished Member of Technical Staff with Bell Laboratories (AT&T and Lucent), where he worked on survivable network design, network optimization, and traffic routing. He is currently with the U.S. Army Armament Research, Development, and Engineering Center, Picatinny Arsenal, NJ. His research interests include scalable lethality, persistent surveillance, weapon–target pairing, and deconfliction.

Dr. Bogdanowicz is the author of numerous journal/conference research papers, and has been a frequent Speaker at academic and industrial events. He has significant contributions to graph theory in open literature.

Antony Tolano received the Master's degree in computer science from East Stroudsburg University of Pennsylvania, East Stroudsburg, upon the completion of his modeling and simulation project and thesis, Battlespace Information Simulator.

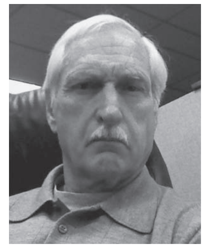
He is currently a Lead Software Engineer with the U.S. Army Armament Research, Development, and Engineering Center, Picatinny Arsenal, NJ.

Mr. Tolano was a recipient of several recognition awards, including Performance Excellence and Dedicated Performance awards.

Ketula Patel received the B.S. degree in computer engineering from New Jersey Institute of Technology, Newark, in 2002, and the M.S. degree in computer engineering, specializing in wireless communications, from Stevens Institute of Technology, Hoboken, NJ, in 2005.

She is currently the Intelligent Systems Branch (ISB) Chief with the U.S. Army Armament Research, Development and Engineering Center (ARDEC), Picatinny Arsenal, NJ. As ISB Branch chief, she manages efforts focused on sensor-to-shooter network lethality technology development and collaborative manned/unmanned effects teaming technologies. Prior to this, she led various research and development programs including radio-frequency localization, denied GPS, and robotics efforts for Army S&T and special operations forces. She is an author or coauthor of published papers in various topics.

Ms. Patel is a member of the Army Acquisition Corps with level 3 certifications in system planning, research, development and engineering. She was a recipient of two Army R&D awards and has coauthored and published papers in various topics.



Norman P. Coleman received the B.A. degree from the University of Virginia, Charlottesville, and the M.A. and Ph.D. degrees from Vanderbilt University, Nashville, TN, all in mathematics.

He is currently a Senior Scientist with the U.S. Army Armament Research, Development, and Engineering Center, Picatinny Arsenal, NJ, specializing in intelligent systems, robotics, controls and decision systems. He has over forty years of government research, development, and management experience. He is a holder of five patents and the author of over a hundred open literature publications.

Dr. Coleman is a recipient of three Army R&D awards.