Toward Rapid and Optimal Strategy for Swarm Conflict: A Computational Game Approach

Tao Zhang, Yiji Zhu, Dongying Ma, Chaoyong Li, *Senior Member*, Xiaodong Wang

Abstract

The decision and control problem for swarm operations is crucial for autonomous military conflict management. In this paper, the underlying decision and control problem is treated as a non-cooperative game problem, in which the underlying target assignment problem is generalized to be a graph-theoretic problem. We introduce an algorithm to seek the desired Nash equilibrium with the help of the parallel maximum weight matching algorithm. Then, we prove that the proposed solution is ϵ-Nash with guaranteed computational efficiency, and is well suited for the swarm conflict. Simulation results verified the effectiveness of the proposed solutions.

Index Terms

Game theory, Nash equilibrium, swarm operation, search algorithm

NOMENCLATURE

Δ	Group strategy space
$\mathcal{M}$	Target assignment result
$\mathcal{T}_i$	Allowable target set for unit i
c_i	Attack vector of unit i, that is, $c_i = [c_{i1},\ ...,\ c_{iN}]^T$
c_{ij}	$c_{ij} = 1$ indicates that the unit i views j as its target and there is an edge e_{ij} connected between them, that is, $c_{ij} \in \mathcal{M}$, and $c_{ij} = 0$ the opposite
J	Target assignment value and also the group payoff
N_B	Number of Blue units
N_R	Number of Red units
U	Group strategy
U^{mo}	Motion input strategy
u_i	Strategy/input of unit i

I. INTRODUCTION

IN a military scenario, decision making plays a crucial role in swarm combat operations. The complexity of factors such as combat unit maneuverability, varied threats from adversaries, and limited resource allocation necessitates the development of a large-scale, high-performance, and multitask decision-making system. However, achieving the desired optimal control solution is challenging due to complex dynamics, constraints, and uncertainties associated with real conflict scenarios [1]. To address this, researchers have explored novel algorithms, including reinforcement learning, distributed learning, and convex optimization [2–4].

Among these research methods, game theory has emerged as a promising approach in various fields due to its effectiveness and superiority. Researchers have employed game theory to design guidance algorithms, investigate rendezvous problems, and address Nash equilibrium (i.e., NE) seeking problems in networked multi-player games [5–7]. The application of game theory to discrete-time decision-making problems in military operations was first introduced by Cruz and Simaan [8–10], in which they modeled a conflict scenario of two opposing forces with nonzero sum dynamic game theory and analyzed the effectiveness of game theory in solving multiteam target assignment problems. Based on such a new direction, subsequent studies extended this method to more complex situations. Shen and Wei [11, 12] et al. investigated a more elaborate setting where two opposing forces and an additional non-neutral civilian group are considered in the game-theoretic model. The technical attraction of this method is mainly motivated by the following advantages:

1) **Rationality:** Entities in a conflict setting are generally rational and can adjust their actions or emotional bias in accordance with the actions of others. Game theory, in that case, is naturally a suitable tool for analyzing the performance of each conflict force.
2) **Objective functions:** The opposing forces generally have specific goals in such a conflict scenario, making it easier to model the objective functions, the associated constraints, and control strategies.

Tao, Yiji and Chaoyong are with College of Electrical Engineering, Zhejiang University, Hangzhou, China, 310027.
Dongying and Xiaodong are with Beijing Institute of Electronic System Engineering, Beijing, 100854, China.
Corresponding author: Chaoyong Li and Xiaodong Wang

3) **Discrete-Time Dynamic Model:** The discrete-time dynamic model greatly facilitated the combination of different models and computer-aided game theoretic analyses [11].

Despite its effectiveness, the command and control method for military operations described above suffers from the shortcoming that it focuses on the large time-span decision-making process of the command system and struggles with high real-time decision-making requirements. To address this limitation, researchers have explored fast computing approaches to enable real-time control and decision making tasks in military operations [13–15]. For example, Zhang et al. proposed a computational NE-seeking algorithm for real-time control tasks [13], while Li et al. developed a fast dimensionality reduction-based matrix game solving algorithm for attack-defense scenarios involving unmanned vehicles [15].

Properly assigning weapons or resources to hostile targets, such as incoming missiles, is essential in military operations. Various allocation algorithms, including the Hungarian method and hybrid algorithms combining genetic algorithms and particle swarm optimization, have been proposed [16–25]. However, these schemes may face challenges regarding running speed, as they tend to solve an optimal result while overlooking that, in many military operations, the optimality of the results does not necessarily have a crucial impact on the conflict outcomes. In practice, many one-on-one target assignment problems can be generalized to the weighted matching problem (MWM), allowing for trade-offs between accuracy and algorithmic performance [26–29]. Notable contributions in this area include the Drake-Hougardy algorithm [27] and the LAM algorithm [28], both achieved a linear running time and 1/2 performance ratio (the result is at least 1/2 of the MWM). Furthermore, Pettie and Sanders extended the augmentation method and provided two linear-time 2/3-approximation algorithms, improving the convergence and accuracy performance [29].

While the aforementioned approximation algorithms have proven effective, they rely on a central unit to deploy algorithms and transmit solutions to all units. In the context of large-scale swarm operations, distributed or parallel algorithms are necessary to overcome the limitations of centralized approaches [30–34]. Thorsen [31] developed a two-phase algorithm using atomic operations involving initial cardinality matching and short augmentations. Still, the limitations of atomic operations can pose challenges when a processor cannot apply a short augmentation. Azad [32] combined the maximum cardinality matching algorithm with the weight-increasing cycles algorithm, bringing the result closer to the optimum. However, this algorithm serves as a highly parallel scheme for solving sparse matrices on distributed memory machines and may not directly apply to military operations.

In this paper, we build upon the pioneering work of [8] and aim to overcome the limitations of existing approaches and propose a graph-matching-based game model specifically designed for large-scale military operations. The main contributions of this paper are as follows:

- **Parallel Computational Game Model:** We formulate a graph-matching-based game model that addresses the target assignment problem in large-scale conflict scenarios. Our model enables rapid and optimal decision-making by introducing a parallel computational framework. To the best of our knowledge, our algorithm achieves faster computation for decision-making and target assignment in large-scale conflict scenarios (typically more than 100 combat units) compared to state-of-the-art work [8, 13, 15, 16].
- **Guaranteed Performance:** We prove that the optimal strategy obtained is an ϵ -NE, leading to a guaranteed performance in terms of computational efficiency and optimality.

This paper is organized as follows. In Section II, we will introduce the basic results of MWM and model the swarm conflict control problem. In Section III, we provide algorithms for solving the NE-seeking problem and the target assignment problem and provide analyses of their performance. Section IV describes the experimental results and explanations. Section V provides conclusions of this paper.

II. Preliminaries and Problem Formulation

A. Preliminary results on weighted bipartite graph

Before proceeding further, we introduce some preliminary results on graph theory. An undirected weighted bipartite graph $\mathcal{G} = (\mathcal{V}_L, \mathcal{V}_R, \mathcal{E}, J)$ can be described by the set of disjoint vertex $\mathcal{V}_L = \{1, 2, ..., n_1\}$ and $\mathcal{V}_R = \{1, 2, ..., n_2\}$, by the set of edges $\mathcal{E} \subseteq \mathcal{V}_L \times \mathcal{V}_R$, and by the weight function $J : \mathcal{E} \to \Re^+$ which assigns a positive weight to each edge of $\mathcal{G}$. The subscript *L* and *R* denotes "left" and "right." The set of neighbors $\mathcal{E}_i \subset \mathcal{E}$ of a left vertex $i \in \mathcal{V}_L$ is the set of all edges associated with i. The weight J_{ij} is the weight of an edge between vertex $i \in \mathcal{V}_L$ and vertex $j \in \mathcal{V}_R$.

The set of all right vertices connected to the left vertex i is defined as $\mathcal{T}_i$, that is,

$$\mathcal{T}_i = \{j \in \mathcal{V}_R | (i, j) \in \mathcal{E}\} \tag{1}$$

A matching $\mathcal{M}$ is a set of the weighted edges chosen in such a way that no two edges share an endpoint. A *maximum weight matching* (MWM) is a matching $\mathcal{M}^*$ such that the sum of weights of all edges in $\mathcal{M}^*$ is the largest among all matchings, that is [35]

$$\mathcal{M}^* = \arg\max_{\mathcal{M} \subset \mathcal{E}} \sum_{e \in \mathcal{M}} J(e) \tag{2}$$

As is well known, albeit its effectiveness in finding optimal solutions, the MWM algorithm does not always come with computational efficiency described by running time. In fact, the MWM algorithm struggles in practical implementations with large-scale graphs. Therefore, a trade-off between time efficiency and optimality of solution is desired in real-world applications, commonly defined as the *approximation maximum weight matching* problem. In order to quantify the quality of the solution, we assume the quality of the solution of an approximation algorithm can be measured by a factor α. Hence, the matching $w(\mathcal{M})$ is called an α-approximate matching if the weight is at least $J(\mathcal{M}) \geq \alpha \cdot J(\mathcal{M}^*)$.

Remark 1. *A matching $\mathcal{M}$ does not necessarily contain all left vertices. If it does, we call it a perfect matching [36]. We will discuss it in Section III.*

Given a perfect matching $\mathcal{M}$ with n left vertices and n right vertices, a *path* $P = \{(i_1, m_{i_1}), (i_2, m_{i_2}), ..., (i_k, m_{i_k})\}$ is a path that the edges are alternatively matched and unmatched. P is called a *k-Path* if $n \leq k$. Furthermore, P is called a *complete k-path* if $n = k$.

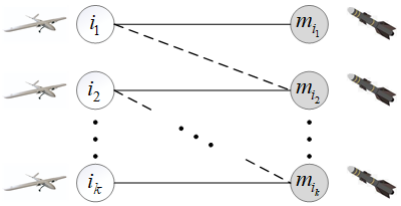
A k-Path P is called a k-Circle $\mathcal{C}$ if $(i_k, m_{i_1}) \in \mathcal{E}$. The k-Path P and the k-Circle $\mathcal{C}$ are shown in Figure 1(a) and Figure 1(b). The symmetric difference between $\mathcal{C}$ and $\mathcal{M}$ is defined as

$$\mathcal{M} \oplus \mathcal{C} \triangleq (\mathcal{M} \backslash \mathcal{C}) \cup (\mathcal{C} \backslash \mathcal{M}) \tag{3}$$

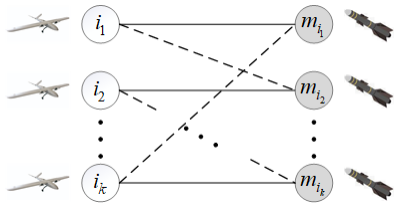
Obviously, $\mathcal{M} \oplus \mathcal{C}$ is also a matching that has the same cardinality as the original matching $\mathcal{M}$. Thus, the gain $g(\mathcal{C})$ of a k-Circle $\mathcal{C}$ is defined as

$$g(\mathcal{C}) \triangleq J(\mathcal{M} \oplus \mathcal{C}) - J(\mathcal{M}) \tag{4}$$

If $g(\mathcal{C}) > 0$, then $\mathcal{C}$ is called an *augmenting circle*, and without loss of any generality, we only consider augmenting circles of the short length, e.g., 2-circle.



(a) k-Path



(b) k-Circle

Fig. 1: k-Path and k-Circle. Matched edges are shown in solid lines, and non-$\mathcal{M}$ edges are shown in dashed lines.

Lemma 1. *Let $\mathcal{M}^*$ be a maximum weighted perfect matching. Then, for any perfect matching $\mathcal{M}$, there exists a set A of disjoint k-Circles such that*

$$J(\mathcal{M} \oplus A) - J(\mathcal{M}) \geq \frac{k}{2k-1}\left(\frac{k-1}{k} J(\mathcal{M}^*) - J(\mathcal{M})\right) \tag{5}$$

Proof. Similar to the proof in [29], we first define a graph $G' = \mathcal{M} \oplus \mathcal{M}^*$ with $2m$ vertices. Then, G' consists of multiple separate circles and paths. Figure 2 shows an example of the circles of G'.

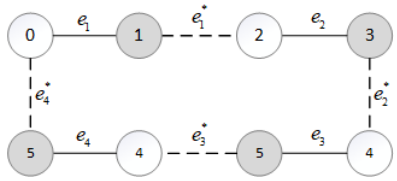


Fig. 2: A circle of G'

Let A_i be the set of $l = \lfloor \frac{m}{k} \rfloor$ disjoint complete k-paths, that is,

$$\begin{aligned} A_i &= \left\{P_i, ..., P_{i+(k-1)(l-1)}\right\} \\ &= \left\{\left\{e_i, ..., e_{i+k-1}\right\}, ..., \left\{e_{i+(k-1)(l-1)}, ..., e_{(i+(k-1)l) \bmod l}\right\}\right\} \end{aligned} \tag{6}$$

It is easy to find that m possible k paths exist when starting at e_i and ending at e_{i+k-1}. Therefore, the number of A_i is $\lfloor \frac{m}{l} \rfloor$ where $\lfloor \frac{m}{l} \rfloor \leq \frac{m}{l}$.

Obviously, among these A_i, the following inequality obviously holds:

$$\max_i g\left(A_i\right) \geq \sum_{i=1}^{\lfloor m/l \rfloor} \frac{g(A_i)}{\lfloor m/l \rfloor} \tag{7}$$

As demonstrated in Figure 1(a), a complete k-path has k $\mathcal{M}$-edges and $k-1$ $\mathcal{M}^*$-edges. Therefore, each edge of $\mathcal{M}$ and $\mathcal{M}^*$ is calculated k times and $k-1$ times, respectively, in the calculation of $\sum \frac{g(A_i)}{\lfloor m/l \rfloor}$, so we have

$$\begin{aligned} \sum_{i=1}^{\lfloor m/l \rfloor} \frac{g(A_i)}{\lfloor m/l \rfloor} &\geq \left\lfloor \frac{m}{k} \right\rfloor \left(\frac{(k-1)\, J(\mathcal{M}^*) - kJ(\mathcal{M})}{m} \right) \\ &= \left\lfloor \frac{m}{k} \right\rfloor \frac{k}{m} \left(\frac{k-1}{k} J(\mathcal{M}^*) - J(\mathcal{M}) \right) \\ &\geq \frac{k}{2k-1} \left(\frac{k-1}{k} J(\mathcal{M}^*) - J(\mathcal{M}) \right) \end{aligned} \tag{8}$$

If we choose $A = \arg\max_i g\left(A_i\right)$, it is clearly that $J(\mathcal{M} \oplus A) - J(\mathcal{M}) \geq \frac{k}{2k-1} \left(\frac{k-1}{k} J(\mathcal{M}^*) - J(\mathcal{M}) \right)$ holds, which completes the proof. □

In this paper, the left vertices of the graph are assumed to be weapons, and the right vertices are targets. From the standpoint of Group 1, its units are the "weapons," and units of Group 2 are the "targets", while the opposite is true for Group 2.

B. Swarm conflict and problem formulation

In a military conflict with a large scale of swarming units such as UAVs or missiles, the swarm conflict among networked combat units can be treated as a two-player/group non-cooperative game problem, with two players/groups termed "Blue" and "Red." Without loss of generality, we assume that Blue is the offensive side and Red is the defensive side. The goal for each group is to collectively maximize the payoff against its adversary, which, by definition, constitutes a non-cooperative game problem for Blue and Red players/groups, and, as is well-established [13], the Nash equilibrium (i.e., NE) of the underlying game captures the optimal solution for both players. Furthermore, it is imperative to maintain the order of each group in order to ensure a maximal impact on the adversary. In other words, optimal task management among combat units is instrumental for the underlying problem, namely, how to assign each combat unit with a specific adversarial target to yield a collectively optimal outcome, given the overall battlefield situation and self-awareness, and how to accomplish this rapidly and optimally is desired for swarming operations.

In what follows, the superscripts 'B' and 'R' identify the belonging, that is, Blue or Red, and are sometimes omitted for simplicity. The main theories and results of this paper are based on the following assumptions:

Assumption 1. ***Target Assignment Constraint***

Any two combat units on the same side cannot attack the same target, which is intuitive and reasonable since locking the same target may cause an unnecessary loss in most cases.

$$\begin{cases} {c_i}^T c_j = 0, \ for\ each\ unit\ i \neq j\ of\ the\ same\ side \\ \sum\limits_{i=1}^{N_R} c_{ij} \in \{0,1\}, j \in \mathcal{V}_B \\ \sum\limits_{i=1}^{N_B} c_{ij} \in \{0,1\}, j \in \mathcal{V}_R \end{cases} \tag{9}$$

where $\mathcal{V}_B = \{1, 2, ..., N_B\}$ *and* $\mathcal{V}_R = \{1, 2, ..., N_R\}$.

Assumption 2. ***Control Input Constraint***

To apply the computational game framework, the control inputs for each unit should be bounded and can be discretized.

In this paper, we propose a novel computational game strategy to tackle the command anc control problem, escalating game theory to large-scale swarming operations and mitigating the otherwise exhaustive computation with a parallel structure, thus potentially offering real-time and optimal target assignment in adversarial scenarios. Note that the option for each unit varies, and the solution is certainly changing over time.

Let u_i be the control input (i.e., strategy) to the ith combat unit consisting of two control types, namely, motion control input u_i^{mo} (e.g., normal accelerations n_i) and target assignment outcome c_i, that is,

$$u_i = u_i^{mo} \times c_i \tag{10}$$

The swarm strategy sets U_B and U_R for Blue and Red is defined as

$$\begin{cases} U_B = U_B^{mo} \times \mathcal{M}_B = \left\{u_1^B, u_2^B, ..., u_i^B, ..., u_{N_B}^B\right\} \\ U_R = U_R^{mo} \times \mathcal{M}_R = \left\{u_1^R, u_2^R, ..., u_i^R, ..., u_{N_R}^R\right\} \end{cases} \tag{11}$$

where U^{mo} is defined as the motion control input and $\mathcal{M}$ is the assignment outcome, that is,

$$\begin{aligned} U^{mo} &= \{u_i^{mo} | i \in \mathcal{V}\} \\ \mathcal{M} &= \{e_{ij} | c_{ij} = 1, i \in \mathcal{V}, j \in \mathcal{T}_i\} \end{aligned} \tag{12}$$

In this paper, we attempt to solve optimal control and decision problem for swarming operations, specifically the underlying target assignment problem, meaning that each agent of the same side has to cooperate within the group and determines the control towards its target. Based on the outcome of the target assignment $\mathcal{M}$ solved by the proposed PHWM algorithm, the combat units decide the motion control u^{mo} and execute the decisions. We aim to develop a rapid strategy to extract the desired U_B and U_R so that the general implications for each combat group are unilaterally maximized. In other words, the desired U_B and U_R constitute a NE for the underlying non-cooperative game problem. In this regard, the problem to be solved in this paper can be formulated as follows [37]

$$\begin{aligned} \textbf{P1:}\ & \textbf{find}\ \ U_B^*, U_R^* \\ & \textbf{s.t.} \begin{cases} J_B\left(U_B^*, U_R^*\right) \geq J_B\left(U_B, U_R^*\right), \forall U_B \in \Delta_B \\ J_R\left(U_B^*, U_R^*\right) \geq J_R\left(U_B^*, U_R\right), \forall U_R \in \Delta_R \end{cases} \end{aligned} \tag{13}$$

where Δ_B and Δ_R are the sets of all admissible decisions.

In problem **P1**, J_B and J_R are payoff functions for Blue and Red, respectively. Then, the desired optimal inputs U_B^* and U_R^* should satisfy the following conditions (Minimax Theorem[38])

$$\begin{cases} \max\limits_{U_B^*} \min\limits_{U_R^*} J_B = \min\limits_{U_R^*} \max\limits_{U_B^*} J_B \\ \max\limits_{U_R^*} \min\limits_{U_B^*} J_R = \min\limits_{U_B^*} \max\limits_{U_R^*} J_R \end{cases} \tag{14}$$

The overall payoff J for the Blue and Red player is the summation of each unit's payoff J_{ij}, which becomes

$$\begin{aligned} J_B(U_B, U_R) &= \sum_{i \in \mathcal{V}_B} J_{ij}^B(u_i^B, u_j^R), j \in \mathcal{T}_i \\ J_R(U_B, U_R) &= \sum_{i \in \mathcal{V}_R} J_{ij}^R(u_i^R, u_j^B), j \in \mathcal{T}_i \end{aligned} \tag{15}$$

where $\mathcal{T}_i$ is defined in (1) representing the set of allowable targets of unit i.

However, solving (15) for NE (i.e., U_B^* and U_R^*) is essentially a matrix game problem. Its calculation, if possible, is shown to be time-consuming and exhaustive [13]. Therefore, a trade-off between the accuracy of the solution and computational solvency must be sought to ensure a timely decision for large-scale swarm operations. Such a solution, albeit sub-optimal, is deemed practical and salient. In this paper, we propose a parallel computation scheme to find a ϵ -Nash solution for the underlying problem and prove that the precision of the solution, captured by ϵ, is uniformly bounded. In this regard, the problem **P1** becomes

$$\begin{aligned} \textbf{P2:}\ & \textbf{find}\ \ \hat{U}_B, \hat{U}_R \\ & \textbf{s.t.} \begin{cases} J_B\left(\hat{U}_B, \hat{U}_R\right) \geq J_B\left(U_B, \hat{U}_R\right) - \epsilon, \forall U_B \in \Delta_B \\ J_R\left(\hat{U}_B, \hat{U}_R\right) \geq J_R\left(\hat{U}_B, U_R\right) - \epsilon, \forall U_R \in \Delta_R \end{cases} \end{aligned} \tag{16}$$

Hence, with problem **P2**, the underlying decision and control problem is successfully converted to a non-cooperative game problem in general and a computational NE-seeking problem in particular, should any combat unit exhibit nonlinear behaviors furthermore, as revealed in (15), the combat unit will first decide which target yields the best collective kill probability and then move accordingly to maximize J_B or J_R. The decision is expected in real-time with updated situational information (e.g., distance, LOS rate). Therefore, how to construct a rapid computation scheme to find the desired NE with acceptable accuracy (i.e., ϵ) is instrumental in swarming conflict management and will be addressed in the next section.

III. Main results

In this section, we proposed the parallel heavy-weight matching algorithm (i.e., PHWM) taken from graph matching theory to assist in solving problem **P2**. As previously established, an analytical solution for a pure NE is impossible due to the intricacy of nonlinear dynamics and the number of combat units. Thus, a computational strategy is in order, and a rapid search algorithm

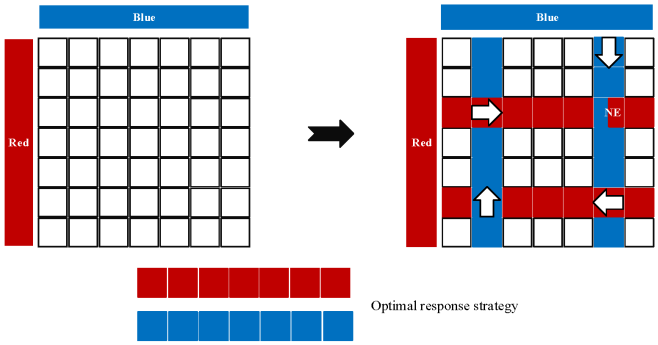


Fig. 3: Action-Reaction Search

becomes instrumental for the underlying problem. To be more precise, the difficulty in solving the matrix game problem grows exponentially with the increase of the operation scale, as specified in [8, 9, 13]. In what follows, we will attempt to integrate the PHWM algorithm into the Action-Reaction Search (i.e., ARS) scheme, whose effectiveness has already been verified in the same venue [13, 39], and actively seek the NE through *optimal response strategy* that alternates between two players, as depicted in Figure 3.

Remark 2. *The optimal response process can be regarded as the target assignment process since J is the assignment value and also the group payoff.*

Algorithm 1 demonstrates the general NE-seeking process working at every decision-making moment. The search process starts with an empty matrix, in other words, a matrix where each element J_{ij} is not calculated yet. Then, beginning with an arbitrary decision of Red or Blue, its opponent makes an optimal response to its decision (Line 7 - Line 11), and the procedure repeats until a NE (if any) is found, that is, Eq. (14) is satisfied.

Algorithm 1 Action-Reaction Search (ARS)

OUTPUT: An ϵ-NE $(\hat{U}_B, \hat{U}_R)$.
BEGIN:

1: By Assumption 2, each combat unit discretizes its motion input, generating the set space of group strategy Δ.
2: **for** every decision-making moment **do**
3: Randomly select a strategy $\tilde{U}_R \in \Delta_R$.
4: Set maximum iteration steps K.
5: Set $k = 0$.
6: **while** $k < K$ **do**
 # Optimal Response Process
7: Apply Eq. (18) to reduce the dimension of the strategy space.
8: Implement the PHWM to find $\hat{U}_B$ such that $J_B\left(\hat{U}_B, \tilde{U}_R\right) \geq J_B\left(U_B, \tilde{U}_R\right), \forall U_B \in \Delta_B$
9: Let $\tilde{U}_B = \hat{U}_B$
 # Optimal Response Process
10: Apply Eq. (18) to reduce the dimension of the strategy space.
11: Implement the PHWM to find $\hat{U}_R$ such that $J_R\left(\tilde{U}_B, \hat{U}_R\right) \geq J_R\left(\tilde{U}_B, U_R\right), \forall U_R \in \Delta_R$
12: **if** $\hat{U}_R == \tilde{U}_R$ **then**
13: An ϵ-NE $(\hat{U}_B, \hat{U}_R)$ is found.
14: Set $k = K$.
15: **else**
16: $\tilde{U}_R = \hat{U}_R$
17: $k = k + 1$
18: **end if**
19: **end while**
20: Apply the control decision $(\hat{U}_B, \hat{U}_R)$.
21: **end for**

END

In particular, a two-player non-cooperative game problem, or swarming conflict management problem to an extent, can be

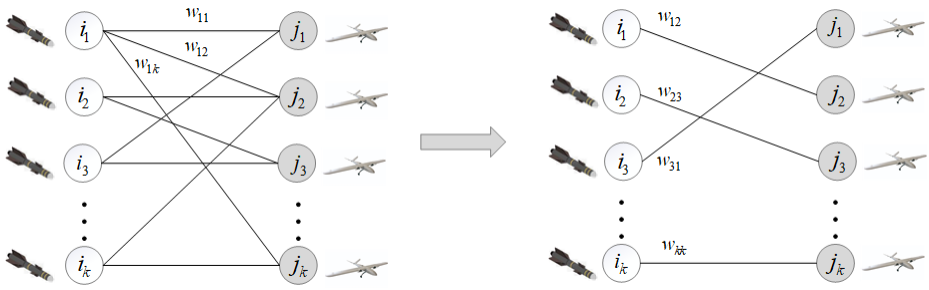


Fig. 4: Decision-making procedure with bipartite graph

conducted over a bipartite graph. In this case, combat units can be treated as two sets of respective and exclusive vertices, and $\mathcal{T}_i$ represents the set of targets that the sensors can observe. If target $j \in \mathcal{T}_i$, we say that unit i has an *edge* e_{ij} connected with target j, and its weight w_{ij} is

$$w_{ij} \in \left\{ J_{ij}\left(u_i^{mo}, u_j\right) | u_i^{mo} \in \Delta_i^{mo} \right\}, j \in \mathcal{T}_i \tag{17}$$

where Δ_i^{mo} is the admissible set of all possible motion control decisions u_i^{mo}. Eq. (17) shows that w_{ij} only relates to unit i's motion inputs since the target assignment input c_i is fixed. Moreover, w_{ij} could be chosen from any value of J_{ij} attributed to the admissible set of the ith unit. Its cardinality depends on the size of the admissible set Δ_i^{mo} of i, as implied in Figure 4.

However, the combination of the edge set will explode during each procedure in ARS due to the ubiquitous choices of w_{ij}, leading to an excessive computational burden. For instance, suppose that $N_B = N_R = 10$ and the size of the motion control input of each unit is $card\left(\Delta_i^{mo}\right) = 100$. We have 100^{10} $\mathcal{G}$ combinations, and a solution to the target assignment will be extremely difficult, if even possible, in a single iteration. In this regard, we select w_{ij} to be the maximal J_{ij}, that is,

$$\begin{aligned} w_{ij}^* &\triangleq \max_{u_i^{mo}} J_{ij}(u_i^{mo}, u_j), j \in \mathcal{T}_i \\ w_{ij} &= w_{ij}^* \end{aligned} \tag{18}$$

Thus, $\mathcal{G}$ will be reduced to $\mathcal{G}^*$ with all other options pruned off.

To verify the reasonableness and effectiveness of (18), let w_{ij}^* and J_{es} be the weight and payoff associated with the optimal search scheme, respectively, commonly conducted with a comprehensive and exhaustive computation scheme where the subscript "es" represents "exhaustive search." Let $\bar{w}_{ij}$ and J_{pr} be the weight and payoff associated with the proposed pruning strategy in (18), respectively, where the subscript "pr" represents "pruning," that is

$$\begin{aligned} J_{es}(U_{es}, \hat{U}) &= \sum_{i \in V} \bar{w}_{ij}, j \in \mathcal{T}_i \\ J_{pr}(U_{pr}, \hat{U}) &= \sum_{i \in V} w_{ij}^*, j \in \mathcal{T}_i \end{aligned} \tag{19}$$

where $\hat{U}$ is the decision set of the adversary. Obviously, through an exhaustive computation, J_{es} will be the maximum result. The following lemma demonstrates that the pruning principle does not evolve at the expense of the optimality of the target assignment decisions.

Lemma 2. *For problem **P2**, suppose graph weight specified in* (17) *with payoffs J_{es} and J_{pr} defined in* (19)*. Then, if $\hat{U}$ is chosen to be the same for J_{es} and J_{pr}, we have $J_{pr} = J_{es}$.*

Proof. Before moving on, we introduce the weight $\bar{w}_{ij}$ as defined in (17) but not necessarily the maximum. Then, we have $\bar{w}_{ij} \leq w_{ij}^*$. For a matching $\mathcal{M}$, let $J_{pr}\left(\mathcal{M}\right)$ be the sum of all w_{ij}^*, and $J_{es}\left(\mathcal{M}\right)$ be the sum of all $\bar{w}_{ij}$, we obtain

$$\begin{aligned} J_{pr}\left(\mathcal{M}\right) &= \sum_{c_{ij} \in \mathcal{M}} w_{ij}^* \\ J_{es}\left(\mathcal{M}\right) &= \sum_{c_{ij} \in \mathcal{M}} \bar{w}_{ij} \end{aligned} \tag{20}$$

It is apparent that for any target assignment outcome (i.e., a matching) $\mathcal{M}$, we have

$$J_{es}\left(\mathcal{M}\right) \leq J_{pr}\left(\mathcal{M}\right) \tag{21}$$

Let $\mathcal{M}_{pr}^*$ and $\mathcal{M}_{es}^*$ be the assignment outcomes associated with the pruning scheme and the exhaustive search scheme, respectively. Without loss of generality, we assume that both schemes are conducted with the PHWM algorithm, and due to the comprehensive nature of the exhaustive search strategy, its matching set contains that of the pruning scheme, that is

$$J_{pr}\left(\mathcal{M}_{pr}^*\right) \leq J_{es}\left(\mathcal{M}_{es}^*\right) \tag{22}$$

In case of $\mathcal{M}_{pr}^{*} = \mathcal{M}_{es}^{*}$, it follows from (21) that

$$J_{es}\left(\mathcal{M}_{es}^{*}\right) \leq J_{pr}\left(\mathcal{M}_{pr}^{*}\right) \tag{23}$$

In case of $\mathcal{M}_{pr}^{*} \neq \mathcal{M}_{es}^{*}$, it is obvious that $\mathcal{M}_{es}^{*}$ is an alternative solution in terms of $\mathcal{G}^{*}$, that is

$$J_{pr}\left(\mathcal{M}_{es}^{*}\right) \leq J_{pr}\left(\mathcal{M}_{pr}^{*}\right) \tag{24}$$

Note that $J_{es}\left(\mathcal{M}_{es}^{*}\right) \leq J_{pr}\left(\mathcal{M}_{es}^{*}\right)$, then we have

$$J_{es}\left(\mathcal{M}_{es}^{*}\right) \leq J_{pr}\left(\mathcal{M}_{pr}^{*}\right) \tag{25}$$

Then, it follows from (22), (23) and (25) that $J_{es}\left(\mathcal{M}_{es}^{*}\right) = J_{pr}\left(\mathcal{M}_{pr}^{*}\right)$. On the other hand, (17) shows that the motion control is attached to an assignment $\mathcal{M}$, then the payoff function J in (42) can be written as the weight of $\mathcal{M}$, which is

$$\begin{array}{l} J_{pr} = J_{pr}\left(\mathcal{M}_{pr}^{*}\right) \\ J_{es} = J_{es}\left(\mathcal{M}_{es}^{*}\right) \end{array} \tag{26}$$

Thus, we have

$$J_{pr} = J_{es} \tag{27}$$

and that completes the proof. □

Invoking *Lemma 2*, it is clear that the pruning scheme does not mitigate the NE of the underlying game. Instead, it will significantly speed up the target assignment process of maximum weighted matching algorithms, for instance, the Kuhn-Munkres (i.e., KM) algorithm, as proved in [13]. However, it should be noted that, albeit its effectiveness, execution of the KM algorithm dictates a centralized agent/node and, hence, makes the computation resources of the overall combat group redundant. In light of the discovery and to make real-time decisions, we propose a parallelized scheme to solve **P2** with the help of the PHWM algorithm and prove that an ϵ-NE can be rapidly ensured with significantly reduced algebraic complexity. *Algorithm 2* demonstrates the working procedure of the PHWM algorithm.

Algorithm 2 Parallel Heavy-Weight Matching (PHWM)

OUTPUT: A target assignment result $\mathcal{M}^{*}$

BEGIN:

1: Apply the PDFM algorithm (*Algorithm 3*) to obtain the initial match $\mathcal{M}_0$

2: With $\mathcal{M}_0$ as input, the PLO algorithm (*Algorithm 4*) is invoked to obtain the target assignment result $\mathcal{M}^{*}$

END

Should *Assumption 1* be satisfied and with *Lemma 2*, a near-optimal assignment result $\mathcal{M}^{*}$ can be expected using the PHWM algorithm as follows:

- Firstly, the Parallel Degree-First Matching (PDFM) algorithm applied for each unit produces an assignment result $\mathcal{M}$. Its procedures are summarized in *Algorithm 3*.
- Secondly, the Parallel Local Optimization (PLO) algorithm takes $\mathcal{M}$ from PDFM as input and produces a near-optimal solution $\mathcal{M}^{*}$. A detailed analysis of PDFM and PLO will be addressed later.

Remark 3. *As mentioned in Remark 1, if a unit (left vertex) is not assigned a target, it can be regarded as redundant at the current decision-making step and choose the target from the last step.*

Now we proceed to analyze the PDFM algorithm implied in *Algorithm 3*. The basic idea is that units with fewer connected edges (i.e., degree) have a higher priority to possess an edge. Moreover, if a unit's degree decreases to 1, this edge will be permanently associated with this particular unit, again ensuring *Assumption 1*. Since every unit prefers an edge with the maximum weight (Line 7), the algorithm will eventually produce an optimal outcome.

The working procedure of *Algorithm 3* is as follows. The PDFM algorithm starts with an empty matching $\mathcal{M} = \emptyset$ shared by every unit. Then, Line 2 to Line 27 of the algorithm runs in parallel on each unit. For unit i, the initial degree d_i is the number of associated edges of unit i, that is, $d_i = |\mathcal{T}_i|$. Then, information exchange is operated via an all-to-all communication among units within the group, during which unit i attempts to claim its largest connected edge by broadcasting it to the group and receives information from its neighbors, and a winner will be declared afterward for the unit with the smallest degree, or the unit with a smaller ID should two units have identical degrees. If unit i wins, it does nothing and prepares for the next iteration. Otherwise, it gives up the edge and decreases its degree by 1. After at most $|\mathcal{T}_i|$ iterations, a matching can be established on all units.

After generating an initial matching with *Algorithm 3*, we may move on to optimize the weight of the matching with the augmenting circles. Intuitively, all augmenting circles of all lengths should be optimized to achieve the best overall outcome,

Algorithm 3 Parallel Degree-First Matching (PDFM)

OUTPUT: An initial matching $\mathcal{M}$.

BEGIN:

1: Let $\mathcal{M} \leftarrow \emptyset$.
2: **for each** unit $i \in \mathcal{V}_L$ **do**
 # The following runs simultaneously on each unit i of the decision-making group.#
3: Let d_i be the number of connected edges, i.e., its initial degree.
4: Let $\mathcal{O}_i \leftarrow \mathcal{T}_i$.
5: Let $success_i \leftarrow false$.
6: **while** $\exists j \in \mathcal{V}$ such that $success_j = false$ **do**
7: Let $m_i = \arg\max_j \{J_{ij} | j \in \mathcal{O}_i\}$ be the target ID that has the maximum weight.
8: Broadcast d_i and m_i to all units.
9: Receive d_j and m_j from all other units $j \in \mathcal{V}$.
10: Let $winner \leftarrow i$.
11: **for each** m_j **do**
12: **if** $m_j = m_i$ **then**
 # We only compare units that have the same expected target as unit i #
13: **if** $(d_j < d_i)$ **or** $(d_j = d_i$ **and** $j < i)$ **then**
 # An unit with a smaller degree wins the target. #
14: Let $winner \leftarrow j$.
15: Break **for** loop.
16: **end if**
17: **end if**
18: **end for**
19: **if** $winner = i$ **then**
 # Win. #
20: $success_i \leftarrow true$
21: **else**
 # Lose. #
22: $d_i \leftarrow d_i - 1$, $\mathcal{O}_i \leftarrow \mathcal{O}_i \backslash m_i$.
23: $success_i \leftarrow false$
24: **end if**
25: Broadcast $success_i$ to all units.
26: **end while**
27: $\mathcal{M} \leftarrow \mathcal{M} \cup (i, m_i)$
28: **end for**

END

but this would be highly time-consuming and counterproductive. Hence, this paper proposes the PLO scheme that focuses on finding a trade-off between accuracy and efficiency.

As illustrated in *Algorithm 4*, each unit within the same group takes a k-circle augmentation approach that optimizes only short-length circles quantified by k, which is chosen to capture the said trade-off. Then they broadcast k-circles and decide which circles to keep. Some k-circles will be dropped, as implied in Line 17, since they contain some edges that are already used for augmentation and marked as visited. As proved in *Theorem 1*, the proposed PLO algorithm ensures a $\frac{2}{3}$-approximation if $k = 3$ is selected.

Lemma 3. *Given a bipartite graph $\mathcal{G}$ with edge matching defined in* (2)*. Then, under Assumption 1 and Algorithms 3 and 4, the PHWM algorithm has an expected running time of $O\left(m \log \frac{1}{\varepsilon} + deg(v)\right)$, where m is the number of edges in $\mathcal{G}$.*

Proof. By *Lemma* 1, let $n = \frac{2k-1}{k}$ and $w^* = \frac{k-1}{k} J(\mathcal{M}^*)$, then we have

$$\mathbb{E}(g(A_i)) \geqslant \frac{1}{n} \left(w^* - J(\mathcal{M})\right) \tag{28}$$

Let $\mathcal{M}_0$ be the initial match constructed from *Algorithm 3* and $\mathcal{M}_i$ represent the matching after augmenting i k-circles, then

$$\mathbb{E}(J(\mathcal{M}_i)) \geqslant \left(1 - e^{-i/n}\right) J^* \tag{29}$$

Algorithm 4 Parallel Local Optimization (PLO)

INPUT: A matching $\mathcal{M}$ from *Algorithm 3*

OUTPUT: A heavy-weight perfect matching $\mathcal{M}$

Set maximum iteration steps K.
Let $it \leftarrow 0$.
Mark all edges of $\mathcal{G}$ as unvisted.
for unit $i \in \mathcal{V}$ **do**
    # The following runs simultaneously on each unit i of the decision-making group.#
    while $it < K$ **do**
        Find an augmenting circle $\mathcal{C}_i$ using depth-first search method [40].
        Broadcast $\mathcal{C}_i$ to all other units.
        Receive $A = \{\mathcal{C}_j | j \in \mathcal{V}\}$ from others.
        if $\forall j \in \mathcal{V}$, $\mathcal{C}_j$ is empty **then**
            # No augmenting circles are found. End iteration. #
            GO TO END.
        else
            for each $\mathcal{C}_j \in A$ in descending order by $g(\mathcal{C}_j)$ **do**
                if $\nexists e \in \mathcal{C}_j$ is visited **then**
                    $\mathcal{M} \leftarrow \mathcal{M} \oplus \mathcal{C}_j$.
                    Mark all edges $e \in \mathcal{C}_j$ as visited.
                else
                    Drop $\mathcal{C}_j$.
                end if
            end for
            $it \leftarrow it + 1$.
        end if
    end while
end for

END

If we set $i = n \ln \frac{1}{\varepsilon}$, then the following inequality can be obtained:

$$\mathbb{E}(J(\mathcal{M}_i)) \geqslant (1-\varepsilon)\, J^* \tag{30}$$

This indicates that if we find $n \ln \frac{1}{\varepsilon}$ short circles and use them for augmenting, we can obtain a $\frac{k-1}{k}$-approximation. Since each left vertex (unit) works in parallel, n/k at most and 1 at least augmenting circle can be used in one iteration.

On the other hand, the average degree of the left vertices is m/n, where m refers to the number of edges in $\mathcal{G}$. Thus, the best-case scenario for the iteration time is

$$\frac{n \ln \frac{1}{\varepsilon}}{n/k} \times \frac{m}{n} = \frac{km}{n} \ln \frac{1}{\varepsilon} \tag{31}$$

and the worst case is $m \ln \frac{1}{\varepsilon}$. Since k is small and $k < n$, the running time complexity of the PLO is $O\left(m \log \frac{1}{\varepsilon}\right)$. The expected running time of the PDFM algorithm is $O\left(\max(\mathcal{T}_i)\right)$, so the total running time of the PHWM becomes

$$O\left(m \log \frac{1}{\varepsilon} + \max(\mathcal{T}_i)\right) \tag{32}$$

which completes the proof. □

Theorem 1. *If Assumption 1 is satisfied and the ARS and the proposed PHWM algorithm are applied in the search for NE, then a $\frac{1}{3}$-NE in **P2** can be found by choosing $k = 3$.*

Proof. Lemma 2 has proved that the pruning method guarantees the invariance of the decision-making process results, so we only need to prove the suboptimality of the PHWM algorithm.

By *Lemma* 1, we have

$$J(\mathcal{M} \oplus A) - J(\mathcal{M}) \geq \frac{k}{2k-1}\left(\frac{k-1}{k} J(\mathcal{M}^*) - J(\mathcal{M})\right) \tag{33}$$

Let $J^* = \frac{k-1}{k} J(\mathcal{M}^*)$, $\mathcal{M}_{n+1} = \mathcal{M}_n \oplus A$, and let $J(\mathcal{M}_n)$ be the weight after n iterations of augmenting, thus we have

$$J(\mathcal{M}_{n+1}) \geqslant \frac{k}{2k-1} J^* + \frac{k-1}{2k-1} J(\mathcal{M}_n) \tag{34}$$

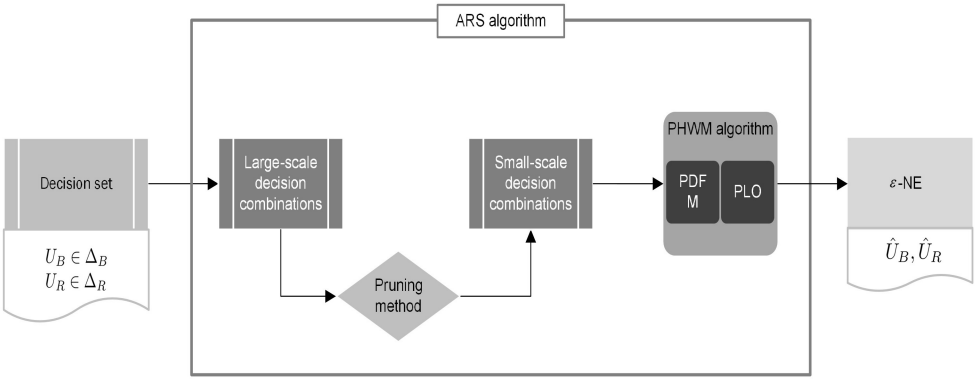


Fig. 5: The overall decision-making procedure

and

$$J(\mathcal{M}_{n+1}) \geq \frac{k}{2k-1}\sum_{i=0}^{n-1}(\frac{k-1}{2k-1})^i J^* + \left(\frac{k-1}{2k-1}\right)^n J(\mathcal{M}_0) \tag{35}$$

As n increases sufficiently large, the resulting weight is

$$J(\mathcal{M}_\infty) \geq \frac{k}{2k-1} \times \frac{1}{1-\frac{k-1}{2k-1}} J^* = J^* = \frac{k-1}{k} J(\mathcal{M}^*) \tag{36}$$

If we choose $n > 1$, the expected weight is the $\frac{2}{3}$-approximation if $k = 3$, which completes the first part of the proposition. As mentioned above, $J = J(\mathcal{M})$, so we have a suboptimal group payoff that satisfies

$$J(\hat{U}) \geq \frac{2}{3} J(U^*), 0 < \epsilon < 1 \tag{37}$$

with $\epsilon = \frac{1}{3}$. Since the individual J is in the range of $[0, 1]$, the above equation turns into

$$J(\hat{U}) \geq J(U^*) - \frac{1}{3} \tag{38}$$

And the NE in **P2** becomes

$$\begin{cases} J_B\left(\hat{U}_B, \hat{U}_R\right) \geq J_B\left(U_B, \hat{U}_R\right) - \frac{1}{3}, \forall U_B \in \Delta_B \\ J_R\left(\hat{U}_B, \hat{U}_R\right) \geq J_R\left(\hat{U}_B, U_R\right) - \frac{1}{3}, \forall U_R \in \Delta_R \end{cases} \tag{39}$$

That completes the proof. □

Remark 4. *Due to the non-uniqueness of the NE of a matrix game, the $\frac{1}{3}$-NE in Theorem 1 is naturally non-unique.*

Figure 5 shows the flow chart of the overall decision-making procedure. At every decision-making step, combat units first discretize their inputs, generating decision sets U_B, U_R. Subsequently, the NE-seeking algorithm, specifically the ARS algorithm, is employed to obtain the desired equilibrium solution. As part of the algorithm, combat units reduce the dimension of the strategy space by employing the pruning method described in Eq. (18) and execute the PHWM algorithm. It is important to highlight that the successful implementation of these algorithms heavily relies on effective communication within the combat units. The information exchange among units occurs with a low communication load, specifically involving the dissemination of degree information (d_i), k-circles ($\mathcal{C}_i$), and their corresponding values ($g(\mathcal{C}_i)$). By considering the computational complexity, the time required to complete a ϵ-NE calculation is determined to be $O\left(Km \log \frac{1}{\varepsilon} + K \max(\mathcal{T}_i)\right)$, where K is the maximum number of iterations set in the ARS algorithm. In practice, the value of K is typically chosen to be not greater than 5, ensuring efficient calculation completion.

IV. Simulation results and discussions

In this section, the effectiveness of the proposed PHWM algorithm is verified with a comprehensive swarm conflict scenario. All simulations are carried out on a desktop computer with a 3.6GHz AMD Ryzen 3500X CPU. We use OpenMP for multi-threading to simulate the implementation of *Algorithm 3* and *Algorithm 4*. Every thread is treated as a combat unit. Without

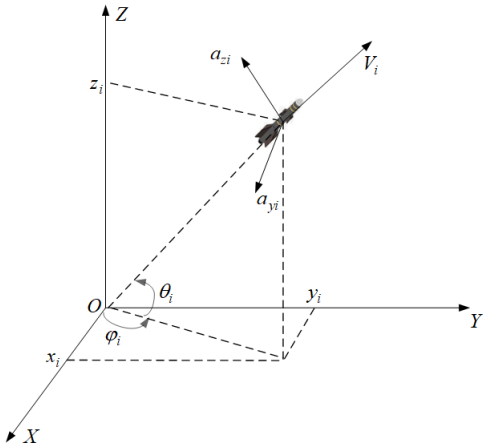


Fig. 6: Dynamics model of combat units

loss of generality, the performance of the proposed algorithm will be evaluated by the average running time of all threads. In what follows, it is assumed that the cardinality of the combat units is identical for both forces, and it is assumed that the Blue force is on the offensive side containing N_B missiles that are targeting a fixed base guarded by the Red force, which in retaliation launches N_R missiles to intercept its Blue counterparts. Therefore, the Red missiles try to hit the Blue ones, while the Blue missiles try to escape them. In our subsequent experiments, we assume that Blue and Red have the same number of combat units and use N to denote the total number of units, that is, $N = N_B + N_R$. Without loss of generality, the movement of the combat unit i is depicted in Figure 6, and its dynamical equations are formulated as follows.

$$\begin{cases} \dot{x}_i = V_i \cos\theta_i \cos\varphi_i \\ \dot{y}_i = V_i \cos\theta_i \sin\varphi_i \\ \dot{z}_i = V_i \sin\theta_i \\ \dot{V}_i = -\frac{D_i}{m_i} - g\sin\theta_i \\ \dot{\theta}_i = -\left(a_{zi} + g\cos\theta_i\right)/V_i \\ \dot{\varphi}_i = \frac{a_{yi}}{V_i \cos\theta_i} \end{cases} \tag{40}$$

where $[x_i, y_i, z_i]$ is the position and V_i, θ_i, φ_i are velocity, pitch and azimuth angles, respectively, m_i is the mass, a_{yi} and a_{zi} are normal accelerations to be determined and we let $n_i = \sqrt{a_{yi}^2 + a_{zi}^2}$. D_i represents the drag force, and

$$D_i = \frac{1}{2} C_d \rho(z_i) S_i V_i^2 \tag{41}$$

where C_d is atmospheric drag coefficient, $\rho(z_i)$ is the atmospheric density at the current altitude, and S_i is the reference area. Therefore, the control input for unit i is $u_i = [a_{yi}, a_{zi}, c_i]$ with c_i defined in (10).

As stated above, the principle for Blue's payoff function is trying to attack and destroy Red's units for Blue's offensive role. Similarly, the payoff function of Red should satisfy the need to minimize the loss caused by Blue. Then, we can set up the payoff functions as follows:

$$\begin{aligned} J_B\left(U_B, U_R\right) &= \sum_{i=1}^{N_B} V_j^R E_{ij}\left(u_i^B, u_j^R\right) \gamma_{ij}^R\left(u_i^B, u_j^R\right), j \in \mathcal{T}_i \\ J_R\left(U_B, U_R\right) &= \sum_{i=1}^{N_R} V_j^B K_{ij}\left(u_j^B, u_i^R\right) \gamma_{ij}^B\left(u_j^B, u_i^R\right), j \in \mathcal{T}_i \end{aligned} \tag{42}$$

where V_j is the engagement value of unit j for unit i and it satisfies $\sum_{j=1}^{N_B} V_j = 1$ and $\sum_{j=1}^{N_R} V_j = 1$. K_{ij} and E_{ij} are the actual probability of kill and the actual probability of evasion,

$$\begin{aligned} K_{ij}\left(u_i, \hat{U}\right) &= PK_{ij}\beta_{ij}^c\left(n_j/n_i, \delta\right), j \in \mathcal{T}_i \\ E_{ij}\left(u_i, \hat{U}\right) &= PE_{ij}\beta_{ij}^c\left(n_j/n_i, \delta\right), j \in \mathcal{T}_i \end{aligned} \tag{43}$$

Under the perfect condition, we define PK_{ij} in (43) as the perfect probability of kill should weapon i attack target j. Similarly, PE_{ij} is the perfect probability of evasion of unit i for weapon j. However, the effectiveness of PK and PE in

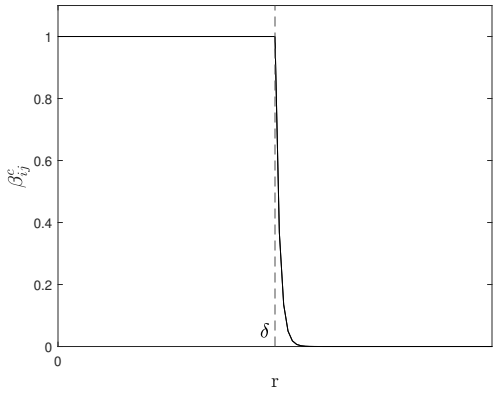


Fig. 7: An example of β_{ij}^c

actual combat scenarios vary according to weapon deficiency. Hence PK_{ij} should be re-calibrated to reflect the instantaneous situations.

In what follows, we consider the maneuverability differences as internal factors. $0 \leq \beta_{ij}^c \leq 1$ is the confidence factor that weapon i successfully attacks target j, $\beta_{ij}^c\left(n_j/n_i, \delta\right)$ is a monotonically decreasing function of the relative orientation and maneuverability differences between i and j, and δ is a critical factor depending on the motion ability. Note that $\beta_{ij}^c \to 1$ if $n_j/n_i \leq \delta$, $\beta_{ij}^c \to 0$ if otherwise. That is, the effectiveness of successful destruction will decrease rapidly when the adversary's maneuverability exceeds its expectation. Specifically, δ can be described by $n_j^{max}/n_i^{max} - \xi$, where ξ is a sufficiently small number, that is, β_{ij}^c decreases significantly as n_j/n_i approaches its maximum, as shown in Figure 7.

In (42), $\gamma_{ij} \in [0, 1]$ is a decision factor for motion control, which is a monotonically decreasing function

$$\begin{aligned} \gamma_{ij}^B\left(u_i, \hat{U}\right) &= \exp\left(\left|\bar{R}_{ij}\right|^2 + \left|\dot{\bar{q}}\right|^2\right) \\ &\quad + \exp\left(-\left|\bar{R}_{i0}\right|^2\right), j \in \mathcal{T}_i \\ \gamma_{ij}^R\left(u_i, \hat{U}\right) &= \exp\left(-\left(\left|\bar{R}_{ij}\right|^2 + \left|\dot{\bar{q}}\right|^2\right)\right), j \in \mathcal{T}_i \end{aligned} \tag{44}$$

where $\hat{U}$ is the adversary's decision set, and $\dot{q}_{ij}\left(u_i, \hat{U}\right)$ is line-of-sight (i.e., LOS) rate and the relative distance $R_{ij}\left(u_i, \hat{U}\right)$ between units i and its intent target j are

$$R_{ij}\left(u_i, \hat{U}\right) = [x_i - x_j, y_i - y_j, z_i - z_j] \tag{45}$$

In our simulation, the individual probability of perfect kill PK_{ij} and the individual destructive value V_j for target j is known to all combat units of each group. V, PK and PE are generated at random. The maneuverability factor $\delta = 2$ in (43). The initial conditions and kill probability for each unit are summarized in Table I.

TABLE I: Initial conditions

Parameters	Red	Blue
Maximal normal acceleration (g)	7	6
Maximal speed (Mach)	1.5	1
Mass (kg)	143	143
Reference area (m^2)	0.02	0.02
Initial speed (Mach)	1.5	1
Initial position (km)	(0,0,0)	(20,20,5)
Midcourse guidance sampling time (sec)	0.5	0.5
Homing guidance sampling time (sec)	0.05	0.05
Homing guidance initiated distance (km)	10	10

We begin the simulation with a simple example to verify the effectiveness of the proposed algorithm in large-scale swarm combat missions. In this case, we assume both forces have 50 missiles, and thus the decision sets are denoted as $U_B = \left\{u_1^B, u_2^B, ..., u_{50}^B\right\}$ and $U_R = \left\{u_1^R, u_2^R, ..., u_{50}^R\right\}$, respectively. The probability of kill PK_{ij} and destructive value V_j are randomly

generated within the range of 0 to 1 and $\sum_{j=1}^{50} V_j = 1$. In addition, without loss of any generality, we assume the target assignment task will only be performed during the midcourse guidance phase. The weapon-to-target designations will be fixed during the homing/terminal guidance since the engagement is relatively short in the homing phase. For a fair comparison, performances of the game-based swarm conflict simulation using the classic centralized maximum weight matching algorithm, Kuhn-Munkres algorithm (i.e., KM) algorithm, and the proposed PHWM algorithm are rigorously studied, and all simulations are conducted with the same initial conditions.

Figure 8 demonstrates the fundamental concept of the proposed strategy regarding calculating the optimal trajectory. Namely, each combat unit will produce a cluster of candidate trajectories copacetic to the size of the decision set. Then an optimal trajectory will be selected rapidly using the proposed PHWM algorithm. Moreover, to examine the proposed algorithm's resiliency against large-scale swarm operations, simulations with a varying number of combat units are performed. As summarized in Table I, the sampling/simulation step for the midcourse and homing guidance is assumed to be 0.5/0.05 second. Noted that the CPU time of KM and our proposed PHWM algorithms refer to the time spent in calculating a single sampling step decision, a comparison of the single step CPU time is plotted in Figure 9, which demonstrates that decision time for the KM algorithm is acceptable in the case that N is relatively small, i.e., $N < 20$. However, it drastically slows down if otherwise and fails to meet the criterion once N exceeds 50. However, the decision time for our proposed parallel algorithm remains steadily rapid as N increases and could still offer real-time decisions in the homing guidance phase even when N approaches 100.

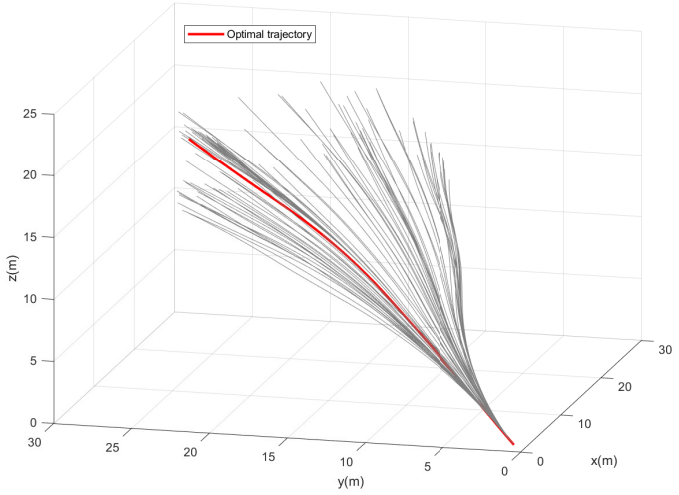


Fig. 8: Candidate trajectories corresponding to the decision set of a Red missile

Figure 10(a) and Figure 10(b) compare the normal acceleration and group payoff value. It is evident that the normal accelerations generated by both schemes are almost identical, while the discrepancies between scaled payoffs are less than

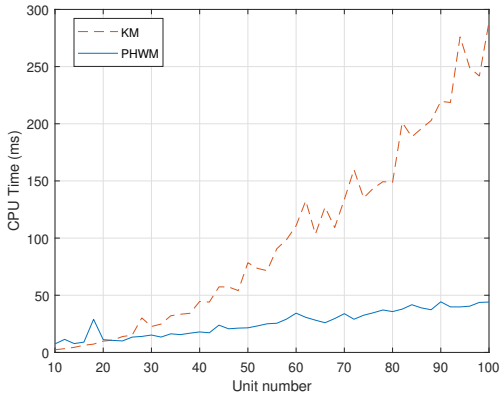
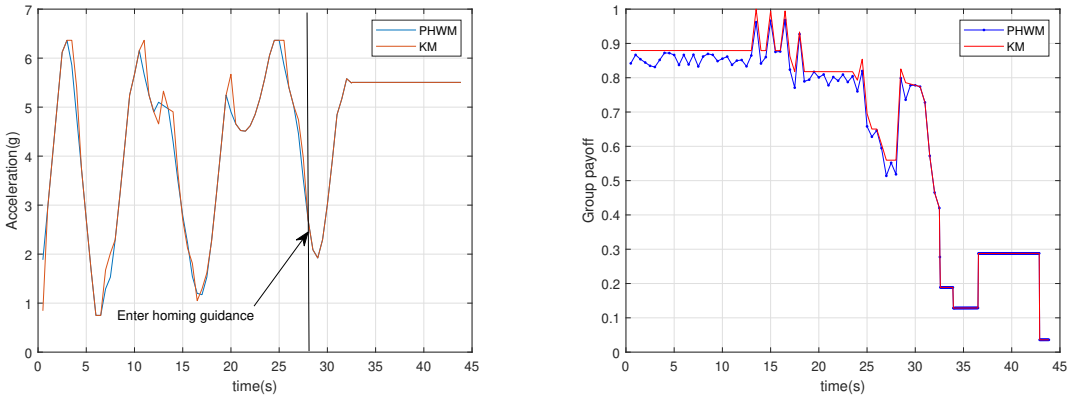


Fig. 9: CPU time

0.04 in value. In other words, the accuracy of the proposed PHWM algorithm is above 95% that of the KM algorithm, which, as previously established, is considered the optimal solution to the underlying problem.



(a) Normal acceleration of a Red missile

(b) Payoff

Fig. 10: Comparison between the KM and PHWM algorithms

In the meantime, the outcomes of air combat scenarios are also examined to validate the effectiveness of the proposed method. In this experiment, we only demonstrate the performance of Red interceptors. Without loss of generality, the blue missiles in the above four examples use game-based decisions to evade interceptors. The miss distances of the red interceptors and blue targets are measured by three miss distance levels (i.e., MDL). Figure 11 demonstrates the outcome of air combat scenarios with different numbers of units N, implying that no matter how N varies, our algorithm can consistently provide excellent interception performance. More specifically, our method can achieve a 100% interception when N is relatively small and can guarantee at least a 70% success rate even in a complex combat scenario. It is worth mentioning that simulations using the KM algorithm with the same experimental conditions have identical results with the PHWM algorithm.

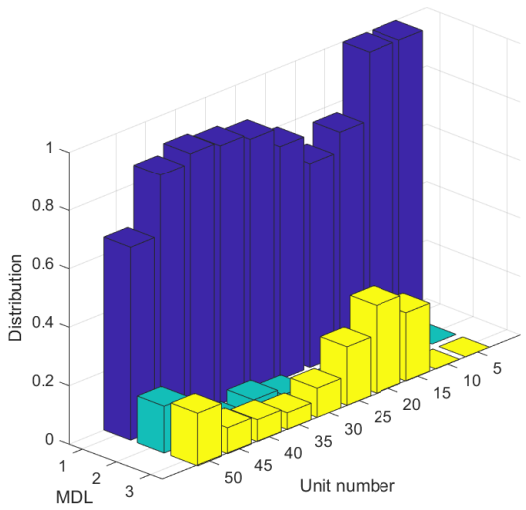


Fig. 11: Distribution of miss distance of Red missile[1]

V. Conclusions

This paper studied decision and control problems for swarm conflict management. We demonstrate that motion control and target assignment among large-scale swarm operations can be treated as a noncooperative computational game problem, and we then propose a parallel computation framework to ensure each unit seeks the Nash equilibrium of the underlying game. Furthermore, we rigorously prove that the proposed conjecture could ensure a ϵ-Nash solution, and its algorithmic complexity is

[1]Miss distance level (MDL) 1: $< 5m$; Level 2: $5 - 10m$; Level 3: $> 10m$

significantly reduced with the parallel matching principle. Simulation results verified that the proposed scheme offers excellent solution accuracy and rapid decision time, paving the way for potential applications in real-time decision making.

ACKNOWLEDGMENTS

The authors thank the associate editor and anonymous reviewers for their constructive comments that improved the paper. This work is supported, in part, by the National Natural Science Foundation of China (Grant No. 62088101 and 12372050), and by the Zhejiang Provincial Natural Science Foundation of China (Grant No. LR20F030003).

REFERENCES

[1] P. Bagul, K. W. Jenkins, and L. Könözsy, "Computational engineering analysis of external geometrical modifications on MQ-1 unmanned combat aerial vehicle," *Chinese Journal of Aeronautics*, vol. 33, no. 4, pp. 1154–1165, 2020.

[2] H. Liu, F. Peng, H. Modares, and B. Kiumarsi, "Heterogeneous formation control of multiple rotorcrafts with unknown dynamics by reinforcement learning," *Information Sciences*, vol. 558, pp. 194–207, 2021.

[3] C.-X. Shi and G.-H. Yang, "Distributed learning over networks: Effect of using historical observations," *IEEE Transactions on Automatic Control*, vol. 65, no. 12, pp. 5503–5509, 2020.

[4] X. Yan and L. He, "Unpowered approach and landing trajectory planning using second-order cone programming," *Aerospace Science and Technology*, vol. 101, p. 105841, 2020.

[5] S. Chen, Y. Yang, D. Ma, X. Wang, K. Li, and C. Li, "Cooperative guidance law with impact angle coordination: A Nash approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 5, pp. 3924–3931, 2022.

[6] Y. Liu, X. Chen, Y. Zhang, and C. Li, "Sample data game strategy for active rendezvous with disturbance rejection," *Aerospace Science and Technology*, vol. 121, p. 107358, 2022.

[7] H. Dong, J. Li, C. Li, and D. Qi, "Continuous-time distributed Nash strategy over switching topologies with gain adaptation," *Systems & Control Letters*, vol. 153, p. 104955, 2021.

[8] J. B. Cruz, M. A. Simaan, G. Aca, H. Jiang, B. Letellier, M. Li, and Y. Liu, "Game-theoretic modeling and control of a military air operation," *IEEE Transactions on Aerospace & Electronic Systems*, vol. 37, no. 4, pp. 1393–1405, 2001.

[9] J. Cruz, J. B., M. A. Simaan, A. Gacic, and Y. Liu, "Moving horizon Nash strategies for a military air operation," *IEEE Transactions on Aerospace & Electronic Systems*, vol. 38, no. 3, pp. 989–999, 2002.

[10] Y. Liu, M. A. Simaan, and J. B. Cruz, "An application of dynamic Nash task assignment strategies to multi-team military air operations," *Automatica*, vol. 39, no. 8, pp. 1469–1478, 2003.

[11] D. Shen, G. Chen, J. B. Cruz, L. S. Haynes, M. Kruger, and E. Blasch, "Game-theoretic modeling and control of military air operations with retaliatory civilians," in *IEEE Aerospace Conference*. IEEE, 2007, pp. 1–10.

[12] M. Wei, G. Chen, J. B. Cruz Jr, L. Hayes, M. Kruger, and E. Blasch, "Game-theoretic modeling and control of military operations with partially emotional civilian players," *Decision Support Systems*, vol. 44, no. 3, pp. 565–579, 2008.

[13] T. Zhang, C. Li, D. Ma, X. Wang, and C. Li, "An optimal task management and control scheme for military operations with dynamic game strategy," *Aerospace Science and Technology*, vol. 115, p. 106815, 2021.

[14] M. Sani, A. Hably, B. Robu, and J. Dumon, "Real-time game-theoretic model predictive control for differential game of target defense," *Asian Journal of Control*, 2023.

[15] S. Li, M. Chen, Y. Wang, and Q. Wu, "A fast algorithm to solve large-scale matrix games based on dimensionality reduction and its application in multiple unmanned combat air vehicles attack-defense decision-making," *Information Sciences*, vol. 594, pp. 305–321, 2022.

[16] D. Galati, Y. Liu, and M. A. Simaan, "A fast algorithm for unit level team resource allocation in a game environment," in *IEEE Conference on Decision and Control*, vol. 3, 2003, pp. 2872–2877.

[17] R. Beard, T. McLain, M. Goodrich, and E. Anderson, "Coordinated target assignment and intercept for unmanned air vehicles," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 6, pp. 911–922, 2002.

[18] H. Lu, H. Zhang, X. Zhang, and R. Han, "An improved genetic algorithm for target assignment, optimization of naval fleet air defense," in *6th World Congress on Intelligent Control and Automation*, vol. 1, 2006, pp. 3401–3405.

[19] X. Zeng, Y. Zhu, L. Nan, K. Hu, B. Niu, and X. He, "Solving weapon-target assignment problem using discrete particle swarm optimization," in *6th World Congress on Intelligent Control and Automation*, vol. 1, 2006, pp. 3562–3565.

[20] L. Babel, "Coordinated target assignment and UAV path planning with timing constraints," *Journal of Intelligent & Robotic Systems*, vol. 94, no. 3, pp. 857–869, 2019.

[21] O. Karasakal, "Air defense missile-target allocation models for a naval task group," *Computers Operations Research*, vol. 35, no. 6, pp. 1759–1770, 2008.

[22] C. Leboucher, H.-S. Shin, P. Siarry, R. Chelouah, S. L. Menec, and A. Tsourdos, "A two-step optimisation method for dynamic weapon target assignment problem," in *Recent Advances on Meta-Heuristics and Their Application to Real Scenarios*, Rijeka, 2013, ch. 5.

[23] J. Chen, B. Xin, Z. Peng, L. Dou, and J. Zhang, "Evolutionary decision-makings for the dynamic weapon-target assignment problem," *Science in China Series F: Information Sciences*, vol. 52, no. 11, pp. 2006–2018, 2009.

[24] E. Munapo, "Development of an accelerating hungarian method for assignment problems," *Eastern-European Journal of Enterprise Technologies*, vol. 4, no. 4-106, pp. 6–13, 2020.
[25] Z. Han, D. Niyato, W. Saad, T. Başar, and A. Hjørungnes, *Game theory in wireless and communication networks: theory, models, and applications*. Cambridge University Press, 2012.
[26] D. E. D. Vinkemeier and S. Hougardy, "A linear-time approximation algorithm for weighted matchings in graphs," *ACM Transactions on Algorithms*, vol. 1, no. 1, pp. 107–122, 2005.
[27] D. E. Drake and S. Hougardy, "A simple approximation algorithm for the weighted matching problem," *Information Processing Letters*, vol. 85, no. 4, pp. 211 – 213, 2003.
[28] R. Preis, "Linear time 1/2-approximation algorithm for maximum weighted matching in general graphs," in *STACS 99*, C. Meinel and S. Tison, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 259–269.
[29] S. Pettie and P. Sanders, "A simpler linear time 2/3- ε approximation for maximum weight matching," *Information Processing Letters*, vol. 91, no. 6, pp. 271–276, 2004.
[30] D. P. Bertsekas, "Auction algorithms." *Encyclopedia of Optimization*, vol. 1, pp. 73–77, 2009.
[31] A. Thorsen, P. Merkey, and F. Manne, "A simple parallel approximation algorithm for maximum weight matching," in *Proceedings of the Third Conference on Partitioned Global Address Space Programing Models*, 2009, pp. 1–6.
[32] A. Azad, A. Buluç, X. S. Li, X. Wang, and J. Langguth, "A distributed-memory algorithm for computing a heavy-weight perfect matching on bipartite graphs," *SIAM Journal on Scientific Computing*, vol. 42, no. 4, pp. C143–C168, 2020.
[33] J.-H. Hoepman, "Simple distributed weighted matchings," *arXiv preprint cs/0410047*, 2004.
[34] M. Sathe, O. Schenk, and H. Burkhart, "An auction-based weighted matching implementation on massively parallel architectures," *Parallel Computing*, vol. 38, no. 12, pp. 595–614, 2012.
[35] G. Chartrand, *Introductory Graph Theory*. Courier Corporation, 1977.
[36] A. Gibbons, *Algorithmic Graph Theory*. Cambridge University Press, 1985.
[37] W. Lin, C. Li, Z. Qu, and M. A. Simaan, "Distributed formation control with open-loop Nash strategy," *Automatica*, vol. 106, pp. 266–273, 2019.
[38] J. v. Neumann, "Zur theorie der gesellschaftsspiele," *Mathematische Annalen*, vol. 100, pp. 295–320, 1928.
[39] D. G. Galati, "Game Theoretic Target Assignment Strategies in Competitive Multi-team Systems," Ph.D. dissertation, University of Pittsburgh, 2005.
[40] S. Even, *Depth-First Search*, 2nd ed. Cambridge University Press, 2011, p. 46–64.

Tao Zhang received the B.Eng. degree and the M.Eng. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 2020 and 2023, respectively. He is currently pursuing a Ph.D. degree in the Department of Electrical Engineering at McGill University, QC, Canada. His research interests include game theory and stochastic control.

Yiji Zhu received the B.Eng. degree in electrical engineering from Nanjing Tech University, Nanjing, China, in 2022. He is currently pursuing a M.Eng. degree in the Department of Electrical Engineering at Zhejiang University, Hangzhou, China. His research interests include dynamics and control of spacecraft and game theory.

Dongying Ma received the Ph.D. degree in precision instrumentation from the Beijing University of Aeronautics and Astronautics, in 2012. He currently works as an engineer in Beijing Institute of Electronic System Engineering. His research interests include control system technology, and artificial intelligence.

Xiaodong Wang received the B.S. degree from the Northwestern Polytechnical University, Xi'an, China, in 2004, and the M.S. degree from the Beijing Institute of Electronic System Engineering, Beijing, China, in 2007. He currently serves as a senior engineer and division director with the Beijing Institute of Electronic System Engineering. His research interests include swarm intelligence, cooperative control and artificial intelligence.

Chaoyong Li (M'09, SM'22) received the Ph.D. degree in aerospace engineering from the Harbin Institute of Technology, Harbin, China, in 2008. He was a postdoctoral scholar and research scientist with the University of Central Florida, Orlando, FL, USA, and Intelligent Fusion Technology Inc., Germantown, MD, USA, respectively. Since 2015, he has been a Research Professor with the College of Electrical Engineering, Zhejiang University. His recent research focuses on distributed control and optimization of multi-agent systems. Dr. Li is a senior member of IEEE and AIAA, and serves as a Regional Editor for Aerospace Science and Technology.