Runtime Analysis of Evolutionary Algorithms for Multi-party Multi-objective Optimization

Yuetong Sun, Peilan Xu, *Member, IEEE*, Wenjian Luo, *Senior Member, IEEE*

***Abstract*—In scenarios where multiple decision-makers operate within a common decision space, each focusing on their own multi-objective optimization problem (e.g., bargaining games), the problem can be modeled as a multi-party multi-objective optimization problem (MPMOP). While numerous evolutionary algorithms have been proposed to solve MPMOPs, most results remain empirical. This paper presents the first theoretical analysis of the expected runtime of evolutionary algorithms on bi-party multi-objective optimization problems (BPMOPs). Our findings demonstrate that employing traditional multi-objective optimization algorithms to solve MPMOPs is both time-consuming and inefficient, as the resulting population contains many solutions that fail to achieve consensus among decision-makers. An alternative approach involves decision-makers individually solving their respective optimization problems and seeking consensus only in the final stage. While feasible for pseudo-Boolean optimization problems, this method may fail to guarantee approximate performance for one party in NP-hard problems. Finally, we propose evolutionary multi-party multi-objective optimizers (EMPMO) for pseudo-Boolean optimization and shortest path problems within a multi-party multi-objective context, maintain a common solution set among all parties. Theoretical and experimental results demonstrate that the proposed $\text{EMPMO}_{\text{random}}$ outperforms previous algorithms in terms of the lower bound on the expected runtime for pseudo-Boolean optimization problems. Additionally, the consensus-based evolutionary multi-party multi-objective optimizer($\text{EMPMO}_{\text{cons}}^{\text{SP}}$) achieves better efficiency and precision in solving shortest path problems compared to existing algorithms.**

***Index Terms*—Multi-party Multi-objective Optimization, Evolutionary Algorithm, Runtime Analysis, Game.**

I. Introduction

A significant proportion of multi-objective optimization problems (MOPs) involve two or more parties, and multi-objective games are one of typical scenarios [1], where multiple parties negotiate within the same decision space to reach a consensus or compromise. For example, UAV path planning requires consensus between government's concerns about urban safety and the enterprise's focus on economic benefits [2], the public facility construction involves the coordination among multiple functional departments [3], and the investment selection for commercial projects requires the evaluation of risks and benefits by various departments [4]. Liu *et al.* [5] defined such problems as multi-party multi-objective optimization problems (MPMOPs), which aim to find a solution set that satisfies or approximates the true Pareto front (PF) of all parties, corresponding to agreement or compromise through negotiation, respectively.

Because each party in an MPMOP holds a separate MOP, multi-party multi-objective evolutionary algorithms (MPMOEAs) naturally evolve from multi-objective evolutionary algorithms (MOEAs). In other words, MPMOEAs are derived from MOEAs by further considering the relationships between multiple parties. Here, MPMOEAs refer to evolutionary algorithms involving M parties, where $M \geq 2$. In particular, when $M = 2$, it is expressed as bi-party multi-objective evolutionary algorithms. The practice of addressing MPMOPs builds on the successful methodologies established in the evolutionary multi-objective optimization (EMO) community. Based on widely acclaimed MOEAs, MPMOEAs have been developed and can be classified into three categories [6]. Dominance-based algorithms extend MOEAs such as the non-dominated sorting genetic algorithm II (NSGA-II) [7], strength pareto evolutionary algorithm (SPEA2) [4,8] , and multi-objective Immune Algorithm (MIA) [2,9] by incorporating multi-party non-dominated sorting techniques [5,10]. In this type of algorithm, each party performs non-dominated sorting of the population with respect to its own set of objectives, giving each individual a level per party. Then these per-party ranks are used jointly to define a multi-party dominance relationship. This allows selecting or ranking individuals in a way that reflects consensus or trade-offs among all parties. Decomposition-based methods leverage frameworks like the multi-objective evolutionary algorithm based on decomposition (MOEA/D) [11]. Chang *et al.* [6] proposed a party-by-party optimization strategy combined with MOEA/D, further refining the approach for solving bi-party multi-objective optimal power flow problems [12]. Indicator-based algorithms, such as those proposed by Song *et al.* [13], integrate indicator-based multi-objective optimization techniques like SMS-EMOA [14] with multi-party non-dominated sorting. Overall, these studies illustrate the development of MPMOEAs, although they remain primarily empirical in nature. However, this also highlights the need for supplementary theoretical analysis.

Experimental results provide empirical support for the favorable performance of MPMOEAs, particularly for MPMOPs that involve common solutions (i.e., the intersection of the

This work has been accepted for publication in IEEE Transactions on Evolutionary Computation. The final published version will be available via IEEE Xplore.

This work is partly supported by the Natural Science Foundation of Jiangsu Province (Grant No. BK20230419), Natural Science Foundation of the Jiangsu Higher Education Institutions of China (Grant No. 23KJB520018), National Natural Science Foundation of China (Grant No. U23B2058).(*Corresponding author: Peilan Xu.*)

Yuetong Sun and Peilan Xu are with School of Artificial Intelligence, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail:202283460028@nuist.edu.cn; xpl@nuist.edu.cn).

Wenjian Luo is with Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Institute of Cyberspace Security, School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518055, Guangdong, China (e-mail:luowenjian@hit.edu.cn).

parties' Pareto-optimal solution sets). To the best of our knowledge, despite significant progress in evolutionary multi-objective optimization [15]–[19], no theoretical analysis currently offers performance guarantees for evolutionary multi-party multi-objective optimization (EMPMO). This raises several key questions: Can traditional MOEAs effectively solve MPMOPs? Are their theoretical guarantees applicable in multi-party contexts? Furthermore, how can a simple and general framework for MPMOPs be designed to facilitate the theoretical analysis of evolutionary algorithms' approximation performance? It is worth noting that for MPMOPs with significant conflicts among parties, the set of common Pareto solutions is often empty, resulting in no common solution. Moreover, there is no clear definition of what constitutes an optimal solution in such cases without a common solution, whereas the case with a common solution is well-defined as the intersection of the individual Pareto fronts of all parties. Therefore, this paper focuses exclusively on scenarios where a common solution exists.

Early practical experience has demonstrated the limitations of MOEAs in addressing MPMOPs [2]. Consider a bi-party bi-objective optimization problem where the Pareto sets of the two parties are denoted as Φ_1 and Φ_2, respectively. The common Pareto set, defined as the intersection of their Pareto-optimal sets, is given by $\Phi_c = \Phi_1 \cap \Phi_2$. For the corresponding four-objective optimization problem, that is, without considering multiple attributes, let its Pareto set be denoted as Φ_3. Clearly, the cardinalities of these sets satisfy $|\Phi_3| \geq \max\{|\Phi_1|, |\Phi_2|\} \geq |\Phi_c|$. This inequality highlights that if the multi-party attributes are ignored, the solutions obtained by the algorithm may include many Pareto-optimal solutions that fail to meet the definition of common Pareto optimality in the multi-party context.

Moreover, applying MOEAs to MPMOPs introduces additional computational overhead. For example, the runtime analysis of simple evolutionary multi-objective optimizer (SEMO) on pseudo-Boolean optimization functions such as COCZ (count ones count zeros) typically involves two stages [20], [21]. The first stage identifies a single Pareto-optimal solution starting from any arbitrary solution, while the second stage involves discovering the complete Pareto set Φ starting from one Pareto-optimal solution. The runtime of the second stage heavily depends on the cardinality of the Pareto set $|\Phi|$. Moreover, in the theoretical analysis of MOEAs, each stage considers all possible non-dominated solutions to compute transition probabilities [22]. When multi-party attributes are ignored, the number of non-dominated solutions in the population grows exponentially, potentially worsening the expected runtime of the algorithm significantly.

Based on the definition of multi-party common Pareto-optimal solutions, a straightforward MPMOEAs can be derived from MOEAs. Taking a bi-party multi-objective optimization problem as an example, we first use MOEAs to obtain the individual Pareto solution sets Φ_1' and Φ_2' for the two parties. The common Pareto solution set is then computed as their intersection $\Phi_c' = \Phi_1' \cap \Phi_2'$. This approach leverages existing multi-objective optimization theories without introducing additional conceptual frameworks. It is evident that if the complete Pareto solution sets are obtained, i.e., $\Phi_1' = \Phi_1$ and $\Phi_2' = \Phi_2$, then $\Phi_c' = \Phi_c$. However, for NP-hard problems, exact solutions cannot be found in polynomial time [23]–[25] unless P=NP. Consequently, this method may result in an empty intersection, making it impossible to identify a solution, or require one party to further compromise on the quality of the solution.

Overall, our contributions are that we propose a series of evolutionary multi-party multi-objective optimizers (EMPMO) for solving MPMOPs (with a focus on the bi-party case for simplicity), in order to fill the theoretical analysis gap between MOEAs and MPMOEAs. These include both an artificial multi-party multi-objective problem and an NP-hard multi-party multi-objective shortest path problem (MPMOSP). We first prove that the transitional algorithm $\text{EMPMO}_{\text{simple}}$ achieves the common Pareto solutions with a runtime bound of $O(3n^2 \log n)$, yet exposes $(1+\varepsilon)$-approximation limitations for NP-hard problems, which is to solve each party's PS separately and then return the intersection. To overcome this, for the artificial problem, we introduce two EMPMOs: $\text{EMPMO}_{\text{random}}$ employs an alternating evolution strategy with the runtime bounded by $O\left(\left(\frac{1}{\varphi} + \frac{1}{1-\varphi}\right)\frac{n^2}{2}\log n\right)$, where φ is the probability of choosing the first party, and $\text{EMPMO}_{\text{payoff}}$ achieves a runtime bound of $O(n \log n)$ for this problem.

For the MPMOSP problem, we demonstrate that it satisfies the multi-party optimal substructure property and design $\text{EMPMO}_{\text{cons}}^{\text{SP}}$, which achieves a $(1+\varepsilon)$-approximation with an expected running time bounded by

$$O\left(n^4 \cdot \delta^{k-1} \cdot \log\left(n\delta^{k-1}\right)\right).$$

where $1+\varepsilon$ represent the approximation ratio, k is the maximum number of objectives among all parties, $w^{\max}$ is the maximum weights across all parties, and $\delta = \frac{n\log(nw^{\max})}{\log(1+\varepsilon)}$.

Experiments confirm that $\text{EMPMO}_{\text{random}}$has near-optimal performance at balanced selection ($\varphi = 0.5$), approaching the performance of$\text{EMPMO}_{\text{payoff}}$. The experiment on a real-world bi-party UAV path planning problem also validate the feasibility of $\text{EMPMO}_{\text{cons}}^{\text{SP}}$and show the limitations of $\text{EMPMO}_{\text{simple}}^{\text{SP}}$.

The rest of this paper is organized as follows. Section II introduces the preliminaries. Section III provides the theoretical analysis for the artificially constructed pseudo-Boolean optimization problem. Section IV also presents the theoretical analysis for the bi-party multi-objective shortest path problem. Section V presents experiments that complement the theoretical analysis. Finally, Section VI concludes the paper and discusses future work.

II. Preliminaries

In this section, we give brief introductions to multi-party multi-objective optimization problems (MPMOPs) and simple evolutionary multi-objective optimizer. In this article, for the pseudo-Boolean optimization problem, we used the number of mutations needed to cover the Pareto front to measure the running time, and for the bi-party multi-objective shortest path problem (BPMOSP), we counted the number of generations.

A. Multi-party Multi-objective Optimization Problems

In real-world scenarios, many optimization problems involve multi-party games, which each party has its own set of objectives that may conflict with those of others. These problems are known as MPMOPs, and represent an extension of classical MOPs, where a single party simultaneously optimizes multiple objectives. In contrast, MPMOPs consider a setting with multiple parties, each having distinct objectives and priorities. Without loss of generality, a minimization MPMOP can be defined as follows:

Definition 1 (MPMOP [5])**.** *Let M denote the number of parties. For each party $m = 1, 2, \ldots, M$, let k_m denote the number of objectives they are concerned with. The MPMOP is defined as:*

$$\min_{\mathbf{x}\in X} \mathcal{F}(\mathbf{x}) = (F_1(\mathbf{x}), F_2(\mathbf{x}), \ldots, F_M(\mathbf{x})), \tag{1}$$

where the objective functions $F_m(\mathbf{x})$ for each party m are given by:

$$F_m(\mathbf{x}) = (f_{m1}(\mathbf{x}), f_{m2}(\mathbf{x}), \ldots, f_{mk_m}(\mathbf{x})), m = 1, 2, \ldots, M, \tag{2}$$

where X is the feasible solution space, $\mathbf{x} \in X$ is the decision vector, bounded by lower and upper limits $x_{\min}$ and $x_{\max}$, and $\mathcal{F}(\mathbf{x}) = (F_1(\mathbf{x}), F_2(\mathbf{x}), \ldots, F_M(\mathbf{x}))$ is the vector of objective functions from all parties.

In multi-objective optimization problems (MOPs) of each party, dominance relations (Definition 2) for minimization problems are introduced to assess solution quality, forming the foundation for defining Pareto optimality (Definition 3).

Definition 2 (Domination)**.** *Let $F_m = (f_1, f_2, \ldots, f_k) : X \to \mathbb{R}^k$ be the objective vector of any party m, where X is the feasible solution space. For two solutions $\mathbf{x}, \mathbf{x}' \in X$:*

1) *$\mathbf{x}$ weakly dominates $\mathbf{x}'$ if $f_j(\mathbf{x}) \leq f_j(\mathbf{x}')$ for all $j \in \{1, \ldots, k\}$, denoted as $\mathbf{x} \succeq \mathbf{x}'$.*
2) *$\mathbf{x}$ dominates $\mathbf{x}'$ if $\mathbf{x}$ weakly dominates $\mathbf{x}'$ and $f_j(\mathbf{x}) < f_j(\mathbf{x}')$ for at least one j, denoted as $\mathbf{x} \succ \mathbf{x}'$.*

Definition 3 (Pareto Optimality)**.** *Let $F_m = (f_1, f_2, \ldots, f_k) : X \to \mathbb{R}^k$ be the objective vector of any party m, where X is the feasible solution space. A solution $\mathbf{x}$ is Pareto optimal if no other solution $\mathbf{x}' \in X$ satisfies $\mathbf{x}' \succ \mathbf{x}$.*

A set $\Phi_m = \{\mathbf{x} \in X \mid \nexists \mathbf{x}' \in X \text{ with } \mathbf{x}' \succ \mathbf{x}\} \subset X$ is called a Pareto set(PS), which contains only all the Pareto optimal solutions on this party. $F_m^* = F_m(\Phi_m)$ is the collection of objective values corresponding to Φ_m and is called Pareto front(PF).

Building on these concepts, the notion of common Pareto optimality is introduced for multi-party multi-objective optimization problems (MPMOPs) (Definition 4).

Definition 4 (Common Pareto Optimality [5])**.** *Let Φ_m represents the Pareto set of party m. For all $m \in \{1, 2, \cdots, M\}$, the set*

$$\Phi = \bigcap_{m=1}^{M} \Phi_m,$$

is referred to as the common Pareto set. Then a solution $\mathbf{x}^* \in \Phi \subset X$ is called common Pareto optimal.

A common Pareto optimal solution represents an equilibrium where the objectives of all parties are simultaneously optimized, rendering it inherently acceptable to all without the need for further compromise. Such a solution is universally optimal across all parties, distinguishing it from non-common solutions, which may satisfy only a subset of the parties' objectives and often require additional negotiations or adjustments to achieve broader consensus. It is worth noting that this paper focuses exclusively on MPMOPs in which common Pareto optimal solutions exist.

B. Simple Evolutionary Multi-objective Optimizer

Simple evolutionary multi-objective optimizer (SEMO) [20] represents the simplest instance of a population-based evolutionary algorithm for multi-objective optimization and is commonly employed in theoretical analyses on multi-objective optimization [26]–[29]. SEMO maintains a population of unfixed size to store all solutions that are not dominated by any other solution encountered thus far. The algorithm begins by randomly selecting an initial solution from the decision space to initialize the population. In each iteration, a solution is randomly chosen from the current population to undergo a one-bit mutation. The resulting mutated solution is then compared against all existing solutions in the population. Dominated solutions are subsequently removed to refine and maintain the population's quality.

Algorithm 1 Simple Evolutionary Multi-objective Optimizer (SEMO)

```
Choose an individual x uniformly from X;
P ← {x};
while termination condition is not met do
    Choose a solution x uniformly at random from P;
    x' is generated by one-bit mutation to x;
    if ∄z ∈ P satisfying (z ≻ x' ∨ f(z) = f(x')) then
        P ← (P \ {z ∈ P | x' ≻ z}) ∪ {x'};
    end if
end while
```

III. Runtime Analysis of Evolutionary Algorithm on Artificial Multi-party Multi-objective Problem

We initiate a theoretical investigation into evolutionary multi-party multi-objective optimizer by employing a pseudo-Boolean function. To the best of our knowledge, this is the first study addressing multi-party multi-objective optimization problems. As the foundation for our analysis, we construct a bi-party bi-objective pseudo-Boolean problem, termed BPAOAZ (bi-party all ones all zeros), tailored to this context. Our work begins by deriving the simple evolutionary multi-party multi-objective optimizer from the well-known SEMO and analyzes its runtime performance. Building on this

baseline, we introduce two variants, i.e., random and payoff-based evolutionary multi-party multi-objective optimizer, inspired by the concepts of "bounded rationality" and "full rationality" from game theory, and provide their respective runtime complexity analysis in solving the BPAOAZ problem. Finally, we demonstrate a runtime performance comparison between these evolutionary bi-party multi-objective optimizers and SEMO, where BPAOAZ is treated as a standard multi-objective optimization problem by the latter. In these algorithms, the population is explicitly or implicitly divided into several subpopulations, each tasked with exploring the Pareto set of a specific decision-making party. Ultimately, the algorithms identify the common Pareto set through an evolutionary process. Therefore, we refer to these algorithms as evolutionary algorithms [30], [31].

A. Artificial Bi-party Bi-objective Optimization Problem

Bi-objective pseudo-Boolean optimization problems, such as LOTZ, COCZ, and OneMinMax, are widely used to analyze the performance of evolutionary multi-objective optimizers [17], [20], [32]. Building upon these foundational bi-objective test cases, we propose an artificial problem referred to as BPAOAZ (bi-party all ones all zeros), designed for two parties, each with two objectives, and ensure the existence of at least one solution in the Pareto optimal set for both parties.

Definition 5 (BPAOAZ)**.** *The pseudo-Boolean function BPAOAZ :* $\{0,1\}^n \to \mathbb{N}^2 \times \mathbb{N}^2$ *is defined as*

$$\mathrm{BPAOAZ}(\mathbf{x}) = (\mathrm{AORZ}(\mathbf{x}), \mathrm{AOFZ}(\mathbf{x}))\,,$$

where $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$*, and* $n = 2a$ *for some* $a \in \mathbb{N}$*. The components* $\mathrm{AORZ}(\mathbf{x})$ *and* $\mathrm{AOFZ}(\mathbf{x})$ *are given as follows:*

$$\mathrm{AORZ}(\mathbf{x}) = \big(f_{11}(\mathbf{x}), f_{12}(\mathbf{x})\big),$$

where

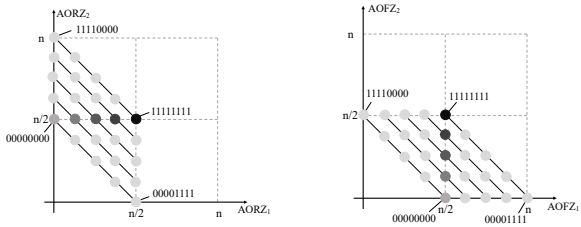
$$f_{11}(\mathbf{x}) = \sum_{i=n/2+1}^{n} x_i, \quad f_{12}(\mathbf{x}) = \sum_{i=1}^{n/2} x_i + \sum_{i=n/2+1}^{n} (1 - x_i);$$

$$\mathrm{AOFZ}(\mathbf{x}) = \big(f_{21}(\mathbf{x}), f_{22}(\mathbf{x})\big),$$

where

$$f_{21}(\mathbf{x}) = \sum_{i=1}^{n/2} (1 - x_i) + \sum_{i=n/2+1}^{n} x_i, \quad f_{22}(\mathbf{x}) = \sum_{i=1}^{n/2} x_i.$$

The problem BPAOAZ consists of two bi-objective pseudo-Boolean optimization problems, AORZ (all ones rear zeros) and AOFZ (all ones front zeros), whose decision space cardinality is 2^n. In AORZ, the first objective is to maximize the number of ones in the second half of a solution, while the second objective is to maximize the sum of the ones in the first half and the zeros in the second half. These objectives are conflicting in the second half of the solution. In AOFZ, the first objective is to maximize the sum of the zeros in the first half and the ones in the second half, and the second objective is to maximize the number of ones in the first half of a solution. These two objectives are conflicting in the first half of the solution. Thus, BPAOAZ involves conflicts between the objectives of the two parties.



(a) AORZ. (b) AOFZ.

Fig. 1: The objective space of the AORZ and AOFZ problem with $n = 8$.

The objective space of AORZ can be partitioned into $\frac{n}{2}+1$ subsets $F_{1,i}$ (Fig. 1a), where $i \in \{0, 1, ..., \frac{n}{2}\}$ represents the number of ones in the first half of the solution. Each subset $F_{1,i}$ contains $\frac{n}{2}+1$ objective vectors of the form $(j, i+\frac{n}{2}-j)$, where $j \in \{0, 1, ..., \frac{n}{2}\}$ is the number of ones in the second half of the solution. Notably, $F_{1,\frac{n}{2}} = \{(\frac{n}{2}, \frac{n}{2}), (\frac{n}{2}-1, \frac{n}{2}+1), ..., (0, n)\}$ represents the PF F_1^*, with cardinality $|F_1^*| = |F_{1,\frac{n}{2}}| = \frac{n}{2}+1$. The subdomains $X_{1,i}$ are defined as the sets of all decision vectors mapped to elements of $F_{1,i}$. The Pareto set $X_1^* = X_{1,\frac{n}{2}} = \{1^{\frac{n}{2}}a, a \in \{0,1\}^{\frac{n}{2}}\}$ has cardinality $|X_{1,\frac{n}{2}}| = 2^{\frac{n}{2}}$. The entire decision space X_1 contains 2^n elements.

The objective space of AOFZ can be partitioned into $\frac{n}{2}+1$ subsets $F_{2,j}$ (Fig. 1b), where $j \in \{0, 1, ..., \frac{n}{2}\}$ represents the number of ones in the second half of the solution. Each subset $F_{2,j}$ contains $\frac{n}{2}+1$ objective vectors of the form $(\frac{n}{2}-i+j, i)$, where $i \in \{0, 1, ..., \frac{n}{2}\}$ is the number of ones in the first half of the solution. Notably, $F_{2,\frac{n}{2}} = \{(\frac{n}{2}, \frac{n}{2}), (\frac{n}{2}+1, \frac{n}{2}-1), ..., (n, 0)\}$ represents the PF F_2^*, with cardinality $|F_2^*| = |F_{\frac{n}{2}}| = \frac{n}{2}+1$. The Pareto set $X_2^* = X_{2,\frac{n}{2}} = \{a1^{\frac{n}{2}}, a \in \{0,1\}^{\frac{n}{2}}\}$ has cardinality $|X_{2,\frac{n}{2}}| = 2^{\frac{n}{2}}$. The entire decision space X_2 contains 2^n elements.

Consider the BPAOAZ problem. Let us take Figure 1a as an example for analysis, with Figure 1b exhibiting similar properties. In Figure 1a, the points that are darker than their surroundings represent the common solutions for each layer, denoted as $X_{1,i} \cap X_{2,i}$, where the solution $\mathbf{x} = (x_1, x_2, \ldots, x_n) \in \{0,1\}^n$ satisfies $\sum_{i=1}^{n/2} x_i = \sum_{i=n/2+1}^{n} x_i$. Due to the dominance relationship between $F_{1,i}$ and $F_{2,i}$ within each layer, the common solutions at a higher layer dominate those at a lower layer for both parties. Specifically, among the five darker points in Figure 1a, the darker points dominate the lighter points for both parties. The darkest point in the figure represents the common Pareto optimal solution. Consequently, there exists only one common Pareto optimal solution for BPAOAZ, which is 1^n. The corresponding common PF is unique and is given by $F^* = \big\{\big(\frac{n}{2}, \frac{n}{2}\big)\big\}$.

B. Runtime Analysis of Evolutionary Multi-party Multi-objective Optimizer

Based on Definition 4 regarding the common Pareto set in MPMOPs, a straightforward approach can be de-

rived from SEMO for solving such problems, referred to as the simple evolutionary multi-party multi-objective optimizer (EMPMO$_{\text{simple}}$), which is outlined in Algorithm 2. The EMPMO$_{\text{simple}}$ operates by independently identifying the Pareto sets for each party and subsequently determining their intersection. Its feasibility depends on two critical assumptions: 1) the presence of at least one solution common to the Pareto optimal sets of both parties and 2) the ability to obtain the complete Pareto set for each evolutionary multi-objective optimizer.

Algorithm 2 Simple Evolutionary Multi-party Multi-objective Optimizer (EMPMO$_{\text{simple}}$)

Randomly select an individual $\mathbf{x} \in X$;
$\Phi \leftarrow \{\mathbf{x}\}$;
for $m \leftarrow 1$ **to** M **do**
    Set initial population $P_m \leftarrow \{\mathbf{x}\}$ for each party;
end for
while termination condition is not met **do**
    for $m \leftarrow 1$ **to** M **do**
        Randomly select an individual $\mathbf{x} \in P_m$;
        Apply one-bit mutation on $\mathbf{x}$ to generate $\mathbf{x}'$;
        if $\nexists \mathbf{z} \in P_m$ **such that** $\mathbf{z} \succ_m \mathbf{x}'$ **or** $F_m(\mathbf{z}) = F_m(\mathbf{x}')$ **then**
            $P_m \leftarrow (P_m \setminus \{\mathbf{z} \in P_m \mid \mathbf{x}' \succ_m \mathbf{z}\}) \cup \{\mathbf{x}'\}$
            if $\nexists \mathbf{z} \in \Phi$ **such that** $\forall m'$ $\mathbf{z} \succ_{m'} \mathbf{x}'$ **or** $F_{m'}(\mathbf{z}) = F_{m'}(\mathbf{x}')$ **then**
                $\Phi \leftarrow (\Phi \setminus \{\mathbf{z} \in \Phi \mid \exists m''\ \mathbf{x}' \succ_{m''} \mathbf{z}\}) \cup \{\mathbf{x}'\}$
            end if
        end if
    end for
end while

The EMPMO$_{\text{simple}}$ algorithm begins by randomly initializing the population for two parties. Initially, the populations of both parties are identical and consist of a single individual. Subsequently, each party employs one-bit mutation to iteratively search for non-dominated solutions, which are stored in their respective populations. Simultaneously, non-dominated solutions common to both parties are identified and stored. $\mathbf{z} \succ_m \mathbf{x}'$ in the algorithm indicates that $\mathbf{z}$ dominates $\mathbf{x}'$ on the objectives of party m, and $\mathbf{z} \succ_{m'} \mathbf{x}'$ in the algorithm indicates that $\mathbf{z}$ dominates $\mathbf{x}'$ on the objectives of party m'.

First, we prove the correctness of the algorithm EMPMO$_{\text{simple}}$, demonstrating that it can identify the common non-dominated solutions of both parties in each iteration and, ultimately, find the common Pareto optimal solution.

Lemma 1. *Consider an algorithm that iteratively constructs the set Φ and the populations P_m, $m \in \{1, \ldots, M\}$ for each party through a sequence of mutation and selection steps, satisfying the following properties:*

1) *A new decision vector is added to P_m if, and only if it is not weakly dominated by any member of P_m, and it is added to Φ if, and only if it is not weakly dominated by any member of Φ across all parties.*
2) *A decision vector is removed from P_m if and only if, a dominating vector is added to P_m, and it is removed from Φ if and only if, a dominating vector is added to Φ for at least one party.*

Then, the set Φ is the common non-dominated solution set of all parties, and Φ is the common Pareto optimal solution set if each population P_m is the Pareto optimal solution set PS_m for party m.

Proof. First, assume a vector $\mathbf{x}$ is added to the set Φ. By Property 1, $\mathbf{x}$ is not weakly dominated by any element in Φ across all parties, and by Property 2, any element dominated by $\mathbf{x}$ is removed from Φ. Thus, Φ contains only $\mathbf{x}$ and non-dominated solutions with respect to each other. According to Algorithm 2, only when x is a non-dominated solution in P_m, that is, a local non-dominated solution, can it be determined whether it can be added to Φ. Therefore, these solutions are also not weakly dominated by any member of P_m. Hence, every element $\mathbf{x}$ in Φ is a non-dominated solution in each party, i.e., $\mathbf{x} \in P_m$ for all m. Therefore, we have $\Phi \subseteq \bigcap_{m=1}^{M} P_m$.

Then, assume there exists a vector $\mathbf{x}'$ such that $\mathbf{x}' \in \bigcap_{m=1}^{M} P_m$ but $\mathbf{x}' \notin \Phi$. Then x' is a non-dominated solution in each P_m. When $P_m = PS_m$, x' is a global non-dominated solution. Since $\mathbf{x}' \in P_m$, it is not weakly dominated by any vector in any population P_m. However, if $\mathbf{x}' \notin \Phi$, this means there exists a vector $\mathbf{z}$ that weakly dominates $\mathbf{x}'$ and has been added to Φ. By Property 1, if $\mathbf{z}$ weakly dominates $\mathbf{x}'$, then $\mathbf{x}'$ cannot be a non-dominated solution for some party m, leading to a contradiction. Thus, we have $\bigcap_{m=1}^{M} P_m \subseteq \Phi$.

From steps 1 and 2, we conclude $\Phi = \bigcap_{m=1}^{M} P_m$. Finally, if each $P_m = PS_m$, then $\Phi = \bigcap_{m=1}^{M} P_m$ represents the common Pareto set. □

The time complexity of EMPMO$_{\text{simple}}$ for solving BPAOAZ primarily arises from independently searching for the complete Pareto optimal sets of AORZ and AOFZ. Finding their non-dominated solutions' intersection adds only a constant-time operation in each iteration. Therefore, the time complexity can be analyzed by referencing [20], which provides a framework for calculating the time complexity of SEMO when solving AORZ and AOFZ, involving two key lemmas, i.e., Lemmas 2 and 3. It is important to note that the running time of the algorithms discussed in this paper is defined as the number of fitness evaluations required to include all Pareto optimal solutions in the population for the first time.

Lemma 2 (General Upper Bound I [20])**.** *Consider an algorithm that iteratively updates a population P via a sequence of mutation and selection steps, with the following properties:*

1) *For each $y \in F \setminus F^*$, the probability that the mutation operator, when applied to any $\mathbf{x} \in X$ with $f(\mathbf{x}) = y$, produces a dominating vector $\mathbf{x}' \succ \mathbf{x}$ is at least $p(y) > 0$, where F is the objective space and F^* represents the PF.*
2) *A new decision vector is added to P if and only if it is not weakly dominated by any member of P.*
3) *A decision vector is removed from P if and only if a dominating vector is added to P.*

Then, the expected number of applications of the mutation operator to non-Pareto-optimal vectors is bounded above by $\sum_{y\in F\setminus F^*} p(y)^{-1}$.

Lemma 3 (General Upper Bound II [20])**.** *Let the dominated part of the decision space* $X\setminus X^*$ *be partitioned into* k *disjoint sets* $X_1, X_2, \ldots, X_k$ *such that* $\bigcup_{i=1}^{k} X_i = X \setminus X^*$ *and* $X_i \cap X_j = \emptyset$ *for* $i \neq j$*. Define the dominance relation on sets as:*

$$X_j \succ X_i \iff \forall (a,b) \in X_j \times X_i : a \succ b$$

Let $d(X_i) := \{X_j \mid X_j \succ X_i\}$ *denote the set of all sets that dominate* X_i*. If the algorithm satisfies the same properties as in Lemma 2, and if* $p(X_i)$ *is a lower bound for the probability that a mutation applied to an individual* $\mathbf{x} \in X_i$ *generates a new individual* $\mathbf{x}'$ *in a dominating decision space subset, i.e.,*

$$0 < p(X_i) \leq \min_{\mathbf{x}\in X_i} \Pr(\mathbf{x}' \in d(X_i) \mid \mathbf{x} \in X_i),$$

then the expected number of times the mutation operator is applied to non-Pareto-optimal decision vectors is bounded above by $\sum_{i=1}^{k} p(X_i)^{-1}$.

Based on the three lemmas above, we can derive the expected running time of $\text{EMPMO}_{\text{simple}}$ for solving the BPAOAZ problem.

Theorem 1. *The expected running time of* EMPMO_{simple} *applied to BPAOAZ is bounded by* $O(3n^2 \log n)$.

Proof. From Lemma 1, it follows that when P_m is the Pareto set of party m, and Φ is the common Pareto set, the running time of Algorithm 2 is equivalent to the cumulative running time.

We begin by calculating the running time to find the Pareto optimal solution set for party 1, whose objective function is $\text{AORZ}(\mathbf{x})$. Without loss of generality, the process is divided into two stages: 1) from initialization to finding a single solution in the Pareto optimal set; 2) from the first Pareto optimal solution to discovering all solutions in the Pareto optimal set.

First, non-Pareto optimal solutions are grouped as $X_{i,j} := \{\mathbf{x} \in X \mid f(\mathbf{x}) = (j, n-i-j)\}$, where $i, j \in \{0, \ldots, n/2\}$, i represents the Hamming distance to the Pareto set, and j is the number of ones in the second half of a solution. By Lemma 2, $X_{i,j}$ forms a partition of the search space. The total number of mutations for non-Pareto-optimal search points is thus bounded by

$$\sum_{j=0}^{n/2}\sum_{i=1}^{n/2} \frac{n}{i} = O(\frac{n^2}{2}\log n).$$

Once a Pareto optimal solution is found, new Pareto optimal solutions can be obtained by flipping bits in the second half of the solution. By expanding from the found Pareto optimal solution toward $j = 0$ and toward $j = \frac{n}{2}$, the complete PF can be found. Let j^* be the f_{11}-value of the first Pareto optimal solution. Define integers $a_t \leq b_t$ such that the population contains Pareto optimal search solutions with an f_{11}-value of i, for all $i \in \{a_t, \ldots, b_t\}$. Initially, $a_t = b_t = j^*$.

While $a_t > 0$, another Pareto optimal solution with f_{11}-value of $a_{t+1} = a_t - 1$ can be obtained by selecting a Pareto optimal solution with f_{11}-value of a_t and making an appropriate 1-bit flip. The probability of that occurring is at least $\frac{1}{\frac{n}{2}+1} \cdot \frac{a_t}{n}$. Summing up expected waiting time and then the time to successfully extend to $a_t = 0$ is

$$\sum_{a_t=1}^{j^*} \frac{n(\frac{n}{2}+1)}{a_t} = O(\frac{n^2}{2}\log j^*).$$

In the worst case, $j^* = \frac{n}{2}$, and the total expected waiting time is $O(\frac{n^2}{2}\log n)$.

While $b_t < \frac{n}{2}$, another Pareto optimal solution with f_{11}-value of $b_{t+1} = b_t + 1$ can be obtained by selecting a Pareto optimal solution with f_{11}-value of b_t and making an appropriate 0-bit flip. The probability of that occurring is at least $\frac{1}{\frac{n}{2}+1} \cdot \frac{\frac{n}{2}-b_t}{n}$. Summing up expected waiting time and then the time to successfully extend to $b_t = \frac{n}{2}$ is

$$\sum_{b_t=j^*}^{\frac{n}{2}-1} \frac{n(\frac{n}{2}+1)}{\frac{n}{2}-b_t} = O(\frac{n^2}{2}\log(\frac{n}{2}-j^*)).$$

In the worst case, $j^* = 0$, and the total expected waiting time is $O(\frac{n^2}{2}\log n)$.

So the total time until both goals have been reached is $O(n^2 \log n)$. Combining the two stages, the expected time for solving AORZ is $O(\frac{3}{2}n^2 \log n)$. By Lemma 1, the total running time of Algorithm 2 for solving BPAOAZ is $O(3n^2 \log n)$ □

The $\text{EMPMO}_{\text{simple}}$ is an idealized algorithm for solving MPMOPs, relying on the assumption that sufficient prior knowledge is available to ensure the algorithm operates correctly. To address this limitation, we propose a more general framework, referred to as the random evolutionary multi-party multi-objective optimizer ($\text{EMPMO}_{\text{random}}$), as shown in Algorithm 3. Unlike $\text{EMPMO}_{\text{simple}}$, $\text{EMPMO}_{\text{random}}$ maintains a single population during its execution period, preserving the negotiation solutions of both parties. In each iteration, a new solution is generated by one party using one-bit mutation. Subsequently, all dominated solutions from that party are removed from the population.

Algorithm 3 Random Evolutionary Multi-party Multi-objective Optimizer ($\text{EMPMO}_{\text{random}}$)

```
Randomly select an individual x ∈ X ;
P ← {x};
while termination condition is not met do
    Randomly select an individual x ∈ P;
    Randomly select a party m;
    Apply one-bit mutation to x to generate x';
    if ∄z ∈ P satisfying (z ≻_m x' ∨ F_m(z) = F_m(x')) then
        P ← (P \ {z ∈ P | x' ≻_m z}) ∪ {x'};
    end if
    P ← {z ∈ P | ∄z' ∈ P \ {z}, z' ⪰_m z};
end while
```

For BPAOAZ, we assume that each party is selected with probabilities φ and $1-\varphi$, respectively, where φ is the probability of choosing the first party. Based on this assumption,

we can derive the expected running time of EMPMO$_{\text{random}}$ for solving the BPAOAZ problem.

Theorem 2. *The expected running time of EMPMO$_{random}$ applied to BPAOAZ is bounded by* $O\left(\left(\frac{1}{\varphi}+\frac{1}{1-\varphi}\right)\frac{n^2}{2}\log n\right)$.

Proof. The evolutionary process is divided into two stages. The first stage aims to find a Pareto-optimal solution in $\Phi_1 \cup \Phi_2$, and the second stage seeks a common Pareto-optimal solution starting from the solution found in the first stage.

Without loss of generality, assume a Pareto-optimal solution in Φ_2 is obtained at the end of the first stage, which is then evolved to obtain the common Pareto set Φ during the second stage. From the analysis of AOFZ, the objective space of party 2 is divided into $1+\frac{n}{2}$ subsets $F_{2,j}$, where $F_{2,\frac{n}{2}} = F_2^*$. Each subset $F_{2,j}$ contains $1+\frac{n}{2}$ objective vectors. According to the evolutionary rules of Algorithm 3, in iteration t, let m_s^t represent the direction of selection and m_m^t represent the direction of actual mutation, which can make $\mathbf{x}' \succ_{m_m^t} \mathbf{x}$. If $m_s^t \neq m_m^t$, both solutions $\mathbf{x}$ and $\mathbf{x}'$ are non-dominated and retained in the population. Conversely, if $m_s^t = m_m^t$, $\mathbf{x}$ will be removed from the population. Therefore, when $m_s^t = m_m^t$, we consider this to be a successful mutation.

Based on the size of $F_{2,j}$, we know that the probability of selecting a solution from $F_{2,j}$ for mutation is at least $\frac{1}{\frac{n}{2}+1}$. During mutation, the probability of flipping a 0 in the second half of a solution is $\frac{\frac{n}{2}-j}{n}$, and the probability of evolving to party 2 in one step is at least $(1-\varphi)\cdot\frac{\frac{n}{2}-j}{n}\cdot\frac{1}{\frac{n}{2}+1}$. Because $X_2^* = \{a1^{\frac{n}{2}} \mid a \in \{0,1\}^{\frac{n}{2}}\}$, $\frac{n}{2}$ successful flips suffice to find a Pareto-optimal solution in Φ_2. Thus, the expected running time for the first stage is

$$\sum_{j=0}^{\frac{n}{2}-1}\frac{n\left(\frac{n}{2}+1\right)}{(1-\varphi)\left(\frac{n}{2}-j\right)} = O(\frac{1}{1-\varphi}\frac{n^2}{2}\log n).$$

In the second stage, starting from the Pareto-optimal solution in Φ_2, the algorithm seeks a common Pareto-optimal solution by evolving on party 1. The probability of selecting this solution from the population and successful mutation for one step is $\frac{\varphi k}{n\cdot(n/2+1)}$, where k is the number of 0 in the first half of the solution. Then the expected time to find the common Pareto set Φ is

$$\sum_{k=1}^{\frac{n}{2}}\frac{n\cdot(\frac{n}{2}+1)}{\varphi k} = O(\frac{1}{\varphi}\frac{n^2}{2}\log n).$$

Combining both stages, the total expected running time is $O\left(\left(\frac{1}{\varphi}+\frac{1}{1-\varphi}\right)\frac{n^2}{2}\log n\right)$. □

EMPMO$_{\text{random}}$, each mutation is controlled by one party at a time. This party is assumed to have incomplete information, specifically the Pareto dominance relationships of solutions relevant to itself, without access to global information. This decision-making model reflects the concept of bounded rationality, as described in game theory, where agents operate with partial knowledge and limited reasoning capabilities [33]. However, inspired by the behavior of fully rational agents, we aim to make optimal decisions in each round, maximizing the decision payoff based on complete information. Therefore, we introduce the concept of multi-party payoff to represent the various possible results in the context of the MPMOPs, such as improvements on both parties, deterioration on both parties, improvement on one party and deterioration on the other party, or no change at all.

Definition 6 (Multi-party Payoff)**.** *Let candidate solutions with the same objective values be grouped into a set X_i. The payoff $\pi_m(X_i, X_j)$ represents the degree of evolution or degeneration for party m when transitioning from set X_i to set X_j. The multi-party payoff $\pi(X_i, X_j)$ on MPMOPs for the transition from X_i to X_j is given by:*

$$\pi(X_i, X_j) = \sum_{m=1}^{M}\pi_m(X_i, X_j).$$

In the specific implementation of this paper, we define $\pi_m(X_i, X_j)$ as the weighted sum of the differences in objective values. Specifically,

$$\pi_m(X_i, X_j) = \begin{cases} 1, & \text{if } \forall k, f_{mk}(X_j) \geq f_{mk}(X_i) \\ -1, & \text{if } \forall k, f_{mk}(X_j) \leq f_{mk}(X_i) \\ 0, & \text{otherwise} \end{cases}$$

where $k \in \{1, ..., k_m\}$ represents the objective, and $m \in \{1, 2\}$ denotes party m.

Consider modeling the evolutionary process using a Markov chain. The corresponding state transition diagram describes the transition probabilities between all states, from the initial state to the target state. Suppose that each edge in the state transition diagram is assigned an additional weight, representing the payoff from transitioning from one state to another. In this case, the payoff $\pi(X_i, X_j)$ from the initial state to the target state should be non-negative. Therefore, we can draw the following conclusion.

Lemma 4. *If the multi-party payoff of each transition between two states, X_i and X_j, is greater than zero, and it is assumed that transitions only occur in the direction of positive payoffs, then no cycle exists between X_i and X_j in the Markov chain.*

Proof. Assume, for the sake of contradiction, that there exists a cycle between states X_i and X_j. This implies the existence of a path that starts at X_i, passes through one or more states, and returns to X_i. Since each transition must occur in the direction of positive payoffs, the total payoff along this cycle must be greater than zero. This contradicts the assumption that the total payoff of any cycle is zero. Therefore, no cycle can exist between X_i and X_j. □

According to Lemma 4, as long as the payoff of each evolutionary step is greater than zero, we can reach the target state from the initial state, since no cycles exist in the evolution chain, preventing the degradation of the population on MPMOPs. Based on this, we propose a payoff-based evolutionary multi-party multi-objective optimizer (EMPMO$_{\text{payoff}}$), as outlined in Algorithm 4. First, the population is initialized with a single individual. The population then undergoes continuous evolution in a loop. In each iteration, a solution $\mathbf{x}$ is randomly selected from the population and mutated to generate a new solution $\mathbf{x}'$. If the payoff from $\mathbf{x}$ to $\mathbf{x}'$ is greater than zero,

it means that $\mathbf{x}$ has evolved towards at least one of the two parties, resulting in $\mathbf{x}'$ that is closer to the common optimal solution. Then $\mathbf{x}'$ is added to the population, and $\mathbf{x}$ is removed. This selection strategy ensures that the population progresses at each step, evolving toward the global Pareto optimal solution.

Algorithm 4 Payoff-based Evolutionary Multi-party Multi-objective Optimizer (EMPMO$_{\text{payoff}}$)

```
Randomly select an individual x ∈ X ;
P ← {x};
while termination condition is not met do
    Randomly select an individual x ∈ P;
    Apply one-bit mutation to x to generate x' ;
    if π_{x,x'} > 0 then
        P ← (P \ {x}) ∪ {x'};
    end if
end while
```

The EMPMO$_{\text{payoff}}$ could achieve more efficient successful mutations when solving the BPAOAZ problem. Consequently, we can derive its expected running time for solving the BPAOAZ problem, as follows:

Theorem 3. *The expected running time of EMPMO$_{payoff}$ applied to BPAOAZ is bounded by $O(n \log n)$.*

Proof. Given the problem setting of BPAOAZ and the mutation strategy, where each successful mutation results in a positive multi-party payoff for all possible state transitions in the evolution process, there is only an objective vector in the population throughout the evolution. Let i represent the number of zeros in the solution. Since the common Pareto optimal solution is $\Phi = 1^n$, it takes at most n steps to increase the number of zeros from i to n, reaching the common Pareto optimal solution. The probability of a solution evolving in each step is $\frac{i}{n}$. Therefore, the total expected running time is given by $\sum_{i=1}^{n} \frac{n}{i}$, which simplifies to $O(n \log n)$. □

EMPMO$_{\text{payoff}}$ can be regarded as a special case of EMPMO$_{\text{random}}$, where the correct party is selected for evolution in each round rather than choosing randomly. Given the specific structure of the problem, where the common Pareto set Φ contains only a single element, the algorithm achieves a runtime bound of $O(n \log n)$. Specifically, the runtime of EMPMO$_{\text{payoff}}$ is equivalent to the time required to compute the transition from 0^n to 1^n via one-bit mutation. Therefore, its time complexity establishes the lower bound $\Omega(n \log n)$ for EMPMO$_{\text{random}}$ in solving the BPAOAZ problem.

Although EMPMO$_{\text{payoff}}$ represents an idealized case, it can be adapted to real-world problems by defining payoffs based on indicators [34] and estimating each party's payoff using historical information [35].

C. Comparison Between Multi-party Multi-objective Optimization and Multi-objective Optimization

Excluding the multi-party attributes, the BPAOAZ problem can be conventionally treated as a general MOP, defined as follows:

Definition 7 (AOAZ)**.** *The pseudo-Boolean function AOAZ, denoted as* $\text{AOAZ} : \{0,1\}^n \to \mathbb{N}^4$*, is defined by*

$$\text{AOAZ}(\mathbf{x}) = (f_{11}(\mathbf{x}), f_{12}(\mathbf{x}), f_{21}(\mathbf{x}), f_{22}(\mathbf{x}))\,,$$

where $f_{11}(\mathbf{x})$, $f_{12}(\mathbf{x})$, $f_{21}(\mathbf{x})$, and $f_{22}(\mathbf{x})$ are defined in Definition 5. Here, $n = 2a$, and $a \in \mathbb{N}$.

The PF F^* of AOAZ is the union of F_1^* of AOFZ and F_2^* of AORZ , with cardinality $|F^*| = n + 1$. The Pareto set $X^* = \{1^{\frac{n}{2}}a, a \in \{0,1\}^{\frac{n}{2}}\} \cup \{a1^{\frac{n}{2}}, a \in \{0,1\}^{\frac{n}{2}}\}$ has cardinality $|X^*| = 2^{\frac{n}{2}+1} - 1$. Moreover, the Pareto set of AOAZ is a superset of the common Pareto set of BPAOAZ.

Once BPAOAZ is treated as a general MOP, the time complexity for finding the complete Pareto set from a single Pareto optimal solution increases significantly. Additionally, the probability of successful evolution from any non-Pareto optimal solution decreases. Example 1 illustrates this scenario.

Example 1. *Consider the following example: Let* $\text{BPAOAZ} : \{0,1\}^{100} \to \{\mathbb{N}^2, \mathbb{N}^2\}$ *with problem size $n = 100$, and* $\text{AOAZ} : \{0,1\}^{100} \to \mathbb{N}^4$ *with the same problem size. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two individuals such that*

$$\text{BPAOAZ}(\mathbf{x}_1) = ((36,44),(56,30))\,,$$
$$\text{BPAOAZ}(\mathbf{x}_2) = ((35,40),(60,25))\,.$$

In the BPAOAZ*, $\mathbf{x}_1$ dominates $\mathbf{x}_2$ on one party, while neither dominates the other on the second party. Therefore, EMPMO can evolve successfully on the first party. However, from the perspective of the* AOAZ*, we have:*

$$\text{AOAZ}(\mathbf{x}_1) = (36,44,56,30)\,,$$
$$\text{AOAZ}(\mathbf{x}_2) = (35,40,60,25)\,.$$

In this case, $\mathbf{x}_1$ and $\mathbf{x}_2$ are mutually non-dominating.

We can further derive the expected running time of using the multi-objective optimization algorithm SEMO to solve AOAZ as follows:

Theorem 4. *The expected running time of SEMO applied to AOAZ is bounded by $O(\frac{5}{2}n^2 \log n)$.*

Proof. Let s denote the Hamming distance between the current solution and the Pareto solution set, representing the minimum number of zeros in either the first or second half of the solution. Let each objective vector constitutes its own subset and the probability of improvement for each set is $\frac{s}{n}$.

The evolution process is divided into two stages. The first stage aims to find a Pareto optimal solution, and the second stage finds the complete Pareto optimal solution set.

For the first stage, we group solutions with the same objective vector together and represent them using the Hamming distance s and i, the number of 1s in the second half of the solution. By Lemma 2, the expected number of mutations required to improve the solution is $\sum_{i=1}^{n/2}\sum_{s=1}^{i}\frac{n}{s} \leq \sum_{i=1}^{n/2} n \log i = n \cdot \log[(\frac{n}{2})!]$. According to Stirling's formula, we have $(\frac{n}{2})! \leq e^{\frac{1}{12}} \cdot \sqrt{\pi n}(\frac{n}{2e})^{\frac{n}{2}}$, and then $\sum_{i=1}^{n/2}\sum_{s=1}^{i}\frac{n}{s} = O(n \cdot \log(n^{\frac{n+1}{2}})) = O(\frac{n^2}{2}\log n)$.

For the second stage, we start from the Pareto optimal solution found and look for the complete Pareto optimal solution set. Without loss of generality, we assume that the

solution found is $0^{\frac{n}{2}}1^{\frac{n}{2}}$. First, we need to flip 0-bit in the first half to get 1^n, and then flip 1-bit in the second half to get the complete Pareto optimal solution set. The maximum number of iterations required is

$$\sum_{i=1}^{\frac{n}{2}} \frac{1}{\frac{1}{n+1}\cdot\frac{i}{n}} + \sum_{j=1}^{\frac{n}{2}} \frac{1}{\frac{1}{n+1}\cdot\frac{j}{n}} = O(2n^2 \log n).$$

Therefore, the total expected running time is $O(\frac{5}{2}n^2 \log n)$. □

In the proofs of Theorem 1 to Theorem 4, we retain the constant terms. Comparing the running time of solving the AOAZ problem and the BPAOAZ problem, we can intuitively find that the EMPMO consistently requires less time. The theoretical upper bound of running time of $\text{EMPMO}_{\text{simple}}$ is longer than that of SEMO because the second stage analysis of $\text{EMPMO}_{\text{simple}}$, when expanding from a Pareto optimal solution to PS, considers the worst-case scenario in both directions. Based on the experimental results in Section V, it's clear that $\text{EMPMO}_{\text{simple}}$ performs better than SEMO. Specifically, the second stage analysis process of SEMO is close to the complete analysis process of $\text{EMPMO}_{\text{random}}$, but the selection probability is smaller. The calculation method of the second stage of SEMO is close to the analysis process of $\text{EMPMO}_{\text{simple}}$, but its selection probability is still smaller. This is because SEMO can't solve BPAOAZ problem directly and it can't achieve good results. SEMO is a standard multi-objective optimization algorithm, whose objective is to solve the complete Pareto optimal solution set. It cannot directly solve multi-party multi-objective optimization problems to obtain the common solution set. A multi-party multi-objective optimization problem can be formulated as a multi-objective optimization problem by introducing suitable trade-offs. However, this will lead to the well-known issue of large PF. This indicates that when the problem inherently involves multiple decision-makers collaboratively making decisions, formulating it as a MPMOP is more efficient than representing it as a standard MOP. Without dividing multiple decision-makers, the problem is multi-objective. Once divided into multiple decision-makers, the solution set of the problem will become smaller, and may even degenerate into a single-objective problem.

Moreover, obtaining the complete Pareto set for a MOP does not guarantee an optimal decision, as each decision-maker focuses solely on their own objectives. For example, in the BPAOAZ problem, only a single common Pareto optimal solution satisfies the requirements of both parties. Therefore, solving the corresponding bi-party multi-objective optimization problem is not only more efficient but also a more rigorous approach.

IV. Runtime Analysis of Evolutionary Algorithms on Bi-party Multi-objective Shortest Path Problem

We analyze the multi-objective shortest path (MOSP) problem as a case study to compare the running times of MP-MOEAs and traditional MOEAs in solving NP-hard problems [22], [36]. MOSP is representative of real-world applications [37], [38], such as UAV path planning, where trade-offs between safety and efficiency must be resolved in multi-party scenarios [2]. This section introduces the bi-party multi-objective shortest path problem (BPMOSP) and derives a baseline algorithm, the simple evolutionary bi-party multi-objective optimizer ($\text{EMPMO}^{\text{SP}}_{\text{simple}}$) , based on the diversity-maintaining evolutionary multi-objective optimizer (DEMO) [22], [39]. We analyze the runtime of this algorithm for solving BPMOSP. Building on this, we propose the consensus-based evolutionary multi-party multi-objective optimizer ($\text{EMPMO}^{\text{SP}}_{\text{cons}}$), which incorporates only candidate solutions achieving consensus between the two parties into the population for updates. A detailed runtime analysis is provided for the proposed algorithm, and its performance is compared with that of traditional evolutionary multi-objective optimizers.

A. Bi-party Multi-objective Shortest Path Problem

We consider the bi-party multi-objective single-source shortest path problem, where the objective is to find the common Pareto shortest path set from a fixed source vertex to each of the remaining target vertices. The formal definition is provided as follows. Assume that all other vertices are reachable from the source vertex directly or indirectly.

Definition 8. *Given a directed weighted graph $G = (V, E, W)$, where V is the set of vertices and $E \subseteq \{(u, v) \in V^2 \mid v \neq u\}$ is the set of directed edges, and $W : E \to \mathbb{R}^{k_1} \times \mathbb{R}^{k_2}$ is the weight function that assigns each edge $e = (u, v) \in E$ a vector of weights corresponding to the objectives of two parties, the bi-party multi-objective single-source shortest path problem (BPMOSP) is defined as:*

$$\min_{\mathbf{x}\in \mathit{Paths}(s,v)} \mathcal{F}(\mathbf{x}) = (F_1(\mathbf{x}), F_2(\mathbf{x}))\,,$$

where $\mathbf{x} = \{v_0, v_1, \ldots, v_l\}$ represents a path from a fixed source vertex $s = v_0$ to any other vertex $v = v_l \in V$, $\text{Paths}(s, v)$ denotes the set of all paths from the source s to target v. The multi-objective functions $F_m(\mathbf{x})$ for each party $m \in \{1, 2\}$, each containing k_m objectives, are given by:

$$F_m(\mathbf{x}) = (f_{m1}(\mathbf{x}), \ldots, f_{mk_m}(\mathbf{x}))\,,$$

where f_{mk} is the sum of the weights along the path $\{v_0, ..., v_l\}$ corresponding to the k-th objective of party m:

$$f_{mk}(\mathbf{x}) = \sum_{j=1}^{l} w_{mk}(v_{j-1}, v_j),$$

with $w_{mk}(v_{j-1}, v_j)$ representing the weight of the edge (v_{j-1}, v_j) corresponding to k-th weight of party m. The path length l satisfies $l \leq n - 1$, where $n = |V|$ is the number of vertices in the graph G.

To facilitate theoretical analysis, we define the following notation:

$$\begin{cases} w_m^{\max} = \max\limits_{k\in\{1,\ldots,k_m\}} \left(\max\limits_{e\in E} w_{mk}(e) \right), \\ w_m^{\min} = \min\limits_{k\in\{1,\ldots,k_m\}} \left(\min\limits_{e\in E} w_{mk}(e) \right), \end{cases}$$

which represent the maximum and minimum weights assigned by party m, respectively. Additionally, we define

$$\begin{cases} w^{\max} = \max\limits_{m\in\{1,2\}} w_m^{\max}, \\ w^{\min} = \min\limits_{m\in\{1,2\}} w_m^{\min}, \end{cases}$$

to denote the maximum and minimum weights across all parties.

In this problem, all edge weights are assumed to be positive. We set $w^{\min} \geq 1$, which can be achieved by normalizing all weights by dividing them by $w^{\min}$. Without loss of generality, among the n vertices, we designate vertex 1 as the source vertex. Thus, the goal is to find the common shortest paths from $s = 1$ to the remaining $n-1$ vertices. According to the problem setting, the objective function satisfies $1 \leq f_{mk}(\mathbf{x}) \leq (n-1)w_m^{\max}$. Then, the dominance relationship between candidate solutions with the same target vertex in the BPMOSP aligns with the definition provided in Definition 2.

The time required to identify the PF for most MOPs is well known to grow exponentially with the problem size [40]. For NP-hard MOPs, approximating the PF is an effective strategy [41], and this approach can also be extended to NP-hard MPMOPs. Let $(1+\varepsilon)$ represent the approximation ratio.

Definition 9 (ε-domination)**.** *Let $\mathbf{x}$ and $\mathbf{x}'$ denote paths with target vertices v_l and $v_{l'}$, respectively. Path $\mathbf{x}'$ weakly ε-dominates path $\mathbf{x}$ for party m, denoted as $F_m(\mathbf{x}') \succeq_{1+\varepsilon} F_m(\mathbf{x})$, if and only if:*

$$\forall k \in \{1,\ldots,k_m\},\ f_{mk}(\mathbf{x}') \leq (1+\varepsilon)\cdot f_{mk}(\mathbf{x}) \quad \textit{and} \quad v_l = v_{l'}.$$

where $\varepsilon > 0$. If $F_m(\mathbf{x}') \succeq_{1+\varepsilon} F_m(\mathbf{x})$ and $F_m(\mathbf{x}') \neq F_m(\mathbf{x})$, then $\mathbf{x}'$ ε-dominates $\mathbf{x}$, denoted as $F_m(\mathbf{x}') \succ_{1+\varepsilon} F_m(\mathbf{x})$. Paths with different target vertices are considered as incomparable.

In the BPMOSP, the concept of common Pareto optimality is consistent with Definition 4 and can be extended to the case of ε-dominance as defined in Definition 9. Assuming the existence of at least one path $\mathbf{x}$ that satisfies the definition of common optimality, this paper focuses on MPMOPs with common solutions.

B. Runtime Analysis of Simple Evolutionary Multi-party Multi-objective Optimizer

Intuitively, Algorithm 2 can also be applied to solve the BP-MOSP. Assuming the existence of a common Pareto solution, the algorithm seeks to identify the complete Pareto set for each party individually, with their intersection yielding the desired common Pareto set. However, for NP-hard problems, obtaining the exact PF is computationally expensive. Therefore, it is often more practical to pursue solutions that satisfy a $(1+\varepsilon)$-approximation ratio, which reduces the time complexity to a polynomial level. As proposed by Horoba [22], a runtime analysis framework for evolutionary algorithms can be used to evaluate their $(1+\varepsilon)$-approximation performance on the multi-objective shortest path problem (MOSP). This framework employs the concept of a box index [41] to partition the objective space.

Definition 10 (Box Index [41])**.** *For the box size $r > 1$, the box index of the objective vector $F_m(\mathbf{x})$, corresponding to the path $\mathbf{x}$, is defined as:*

$$b_r(F_m(\mathbf{x})) = (\lfloor \log_r(f_{m1}(\mathbf{x}))\rfloor, \cdots, \lfloor \log_r(f_{mk_m}(\mathbf{x}))\rfloor).$$

Each box can accommodate at most one individual in the population, effectively controlling the population size through box division and the dominance relationship between boxes. According to the properties of the MOSP, box indices are comparable only when the target vertices of the paths are identical. If the target vertices differ, the box indices are considered incomparable. Formally, given two paths $\mathbf{x}$ and $\mathbf{x}'$ with the identical target vertices, $b_r(F_m(\mathbf{x}')) \succeq b_r(F_m(\mathbf{x}))$ if and only if

$$\forall k \in \{1,\ldots,k_m\},\ \lfloor \log_r(f_{mk}(\mathbf{x}'))\rfloor \leq \lfloor \log_r(f_{mi}(\mathbf{x}))\rfloor,$$

and $b_r(F_m(\mathbf{x}')) \succ b_r(F_m(\mathbf{x}))$ if and only if

$$\begin{aligned} &\forall k \in \{1,\ldots,k_m\}, \lfloor \log_r(f_{mk}(\mathbf{x}'))\rfloor \leq \lfloor \log_r(f_{mk}(\mathbf{x}))\rfloor, \\ &\exists k \in \{1,\ldots,k_m\}, \lfloor \log_r(f_{mk}(\mathbf{x}'))\rfloor < \lfloor \log_r(f_{mk}(\mathbf{x}))\rfloor. \end{aligned}$$

Using the runtime analysis framework proposed by Horoba [22], we independently search for $(1+\varepsilon)$-approximation solutions for both parties. However, unlike the conclusion in Section III-B, after introducing approximation analysis, there is no guarantee that the algorithm can find a common Pareto set satisfying the $(1+\varepsilon)$-approximation ratio.

Lemma 5. *Let Φ_1 and Φ_2 represent the Pareto-optimal solution sets of the BPMOSP for two parties, respectively, and assume there exists a common Pareto solution $\mathbf{x}^* \in \Phi_1 \cap \Phi_2$. Suppose $\Phi_1^{\varepsilon_1}$ and $\Phi_2^{\varepsilon_2}$ are the Pareto solution sets with approximation ratio $(1+\varepsilon_1)$ and $(1+\varepsilon_2)$, respectively, obtained via EAs. Then, there exists a non-zero probability that the intersection of these approximate sets is empty, i.e., $Pr\left(\Phi_1^{\varepsilon_1} \cap \Phi_2^{\varepsilon_2} = \emptyset\right) > 0$.*

Proof. Assume that a common Pareto solution $\mathbf{x}^*$ exists. According to Definition 10, the box indices for party 1 and party 2 corresponding to $\mathbf{x}^*$ are $b_{r_1}(F_1(\mathbf{x}^*))$ and $b_{r_2}(F_2(\mathbf{x}^*))$, where $r_1 = 1+\varepsilon_1$ and $r_2 = 1+\varepsilon_2$ represent the approximation ratios for each party. Since evolutionary algorithms (EAs) are employed to approximate the Pareto sets, there exist solutions $\mathbf{x}_1$ and $\mathbf{x}_2$ such that:

$$b_{r_1}(F_1(\mathbf{x}_1)) = b_{r_1}(F_1(\mathbf{x}^*)), \quad b_{r_2}(F_2(\mathbf{x}_2)) = b_{r_2}(F_2(\mathbf{x}^*)).$$

Given that each approximation box can contain at most one solution, it follows that $\Phi_1^{\varepsilon_1} = \{\mathbf{x}_1\}$ and $\Phi_2^{\varepsilon_2} = \{\mathbf{x}_2\}$. Without loss of generality, we aim to prove that at least one scenario exists where $\mathbf{x}_1 \neq \mathbf{x}_2$.

To illustrate this, consider a weighted directed graph $G = (V, E)$, as depicted in Fig. 2, where $V = \{1,2,3,4,5\}$. Let $\varepsilon_1 = \varepsilon_2 = 1$, resulting in a final box size of 2 for both parties. Table I enumerates all possible paths from the source vertex 1 to the target vertex 5 within the solution space, along with their corresponding objective values for the two parties. Boldface values highlight objective values that are not dominated by other solutions.

For party 1, the Pareto set from vertex 1 to vertex 5 is $\Phi_1 = \{(1,3,5),(1,3,4,5)\}$. For party 2, the Pareto set is

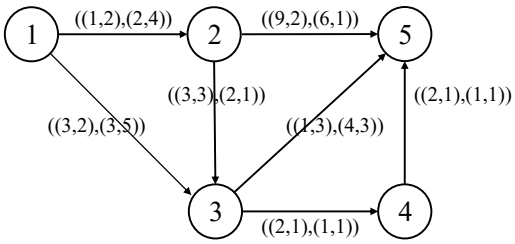


Fig. 2: The weighted directed graph G.

TABLE I: All possible solutions from vertex 1 to vertex 5 and their respective objective values on the two parties.

Paths	Objective value on party 1	Objective value on party 2
$(1,2,5)$	$(10,4)$	$(\mathbf{8},\mathbf{5})$
$(1,2,3,5)$	$(5,8)$	$(8,8)$
$(1,2,3,4,5)$	$(8,7)$	$(6,7)$
$(1,3,5)$	$(\mathbf{4},\mathbf{5})$	$(7,8)$
$(\mathbf{1},\mathbf{3},\mathbf{4},\mathbf{5})$	$(\mathbf{7},\mathbf{4})$	$(\mathbf{5},\mathbf{7})$

$\Phi_2 = \{(1,2,5),(1,3,4,5)\}$. Therefore, $\mathbf{x}^* = (1,3,4,5)$ is the common Pareto-optimal solution.

Now, consider the approximate solutions obtained via the evolutionary algorithm. Let the path sets for the two parties be $P_1 = \{(1,3,5),(1,3,4,5)\}$ for party 1, and $P_2 = \{(1,2,5)\}$ for party 2. The corresponding box indices for party 1 and party 2 are $b_{r_1}(F_1(P_1)) = \{(2,2)\}$ and $b_{r_2}(F_2(P_2)) = \{(3,2)\}$.

For the solutions in P_1, the box indices for party 2 are $\{(2,3),(2,2)\}$, which are not included in $b_{r_2}(F_2(P_2))$. This implies that the intersection of $\Phi_1^{\varepsilon_1}$ and $\Phi_2^{\varepsilon_2}$ is empty. Thus, despite the existence of a common Pareto solution $(1,3,4,5)$, we cannot find it in the approximate solution sets at this time.

Therefore, the probability that the intersection of the approximate solution sets is empty is non-zero: $\Pr(\Phi_1^{\varepsilon_1} \cap \Phi_2^{\varepsilon_2} = \emptyset) > 0$. □

However, as an approximation algorithm, this issue can be mitigated by relaxing the approximation ratio of one party. Taking Fig. 2 as an example, we relax the box size of party 2, represented as $r_2' = 1+\varepsilon_2'$, until a solution is found within the common non-dominated box, setting $\varepsilon_2' = 2$ expands the box size for party 2, resulting in $b_{r_2'}(F_2(P_2)) = \{(1,1)\}$. The box indices of the solutions in P_1 for party 2 are $(1,1)$. At this point, the path $(1,3,5)$ and $(1,3,4,5)$ satisfies the required conditions and is output as the final solutions from vertex 1 to vertex 5.

The pipeline of EMPMO$^{\text{SP}}_{\text{simple}}$ for the MPMOSP is outlined in Algorithm 5 and consists of two stages. In the first stage, each party independently obtains its approximate Pareto solution set, denoted as $\Phi_1^{\varepsilon_1}$ and $\Phi_2^{\varepsilon_2}$. In the second stage, under a relaxed approximation policy, one party, assumed here to be party 1, proposes its approximate Pareto solutions $\mathbf{x} \in \Phi_1^{\varepsilon_1}$ sequentially. Party 2 then decides whether to relax its approximation ratio ε_2 and accept the proposed solution as a common solution. This interaction is modeled as an ultimatum game [42].

Algorithm 5 Simple Evolutionary Multi-party Multi-objective Optimizer (EMPMO$^{SP}_{\text{simple}}$)

for $m \leftarrow 1$ **to** M **do**
  Set initial population $P_m \leftarrow \{(1)\}$ for each party;
end for
while termination condition is not met **do**
  for $m \leftarrow 1$ **to** M **do**
    Randomly select an individual $\mathbf{x} \in P_m$;
    Mutate $\mathbf{x}$ to get $\mathbf{x}'$;
    if $\nexists \mathbf{z} \in P_m$ such that $F_m(\mathbf{z}) \succ F_m(\mathbf{x}') \vee b_r(F_m(\mathbf{z})) \succ b_r(F_m(\mathbf{x}'))$ **then**
      $P_m \leftarrow (P_m \setminus \{\mathbf{z} \in P_m \mid b_r(F_m(\mathbf{x}')) \succeq b_r(F_m(\mathbf{z}))\}) \cup \{\mathbf{x}'\}$;
    end if
  end for
end while
for $i \leftarrow 1$ **to** $\mid P_1 \mid$ **do**
  Calculate the minimum $\varepsilon_{2,i}$ corresponding to $P_1(i,:)$
  if $\varepsilon_{2,i} \leq \varepsilon_2^{max}$ **then**
    $\varepsilon'_{2,i} = \varepsilon_{2,i}$
  end if
end for
Find the minimum $\varepsilon'_{2,i}$ and its corresponding solution for each endpoint in P_1
All solutions found constitute the final solution set

Definition 11 (Ultimatum Game)**.** *For BPMOSP, the process of obtaining an approximate common Pareto set from the two parties' approximate Pareto sets, $\Phi_1^{\varepsilon_1}$ and $\Phi_2^{\varepsilon_2}$, is modeled as an ultimatum game $\{\mathcal{M},\Omega,\mathcal{U}\}$, where:*

- *$\mathcal{M} = \{1,2\}$ denotes the set of decision-makers (parties), with party 1 acting as the proposer and party 2 as the responder.*
- *$\Omega = \{\Omega_1,\Omega_2\}$, where $\Omega_1 = Paths(s,v)$ is the policy space of party 1, and $\Omega_2 = \{Accept, Reject\}$ is the policy space of party 2.*
- *$\mathcal{U} = (u_1,u_2)$ represents the utility functions of the two parties. For party 1, the utility function is:*

$$u_1(p,y) = \begin{cases} U_1^{\max}, & \text{if } p \in \Phi_1^{\varepsilon_1} \text{ and } y = Accept, \\ 0, & \text{otherwise}, \end{cases}$$

where $U_1^{\max}$ is the maximum utility when $p \in \Phi_1^{\varepsilon_1}$. For party 2, the utility function is:

$$u_2(p,y) = \begin{cases} U_2^{\max}, & \text{if } p \in \Phi_2^{\varepsilon_2} \\ & \text{and } = Accept, \\ U_2^{\max}\frac{\varepsilon_2^{\max}-\varepsilon_2'}{\varepsilon_2^{\max}-\varepsilon_2}, & \text{if } p \in \Phi_2^{\varepsilon_2^{\max}}, p \notin \Phi_2^{\varepsilon_2} \\ & \text{and } y = Accept, \\ 0, & \text{otherwise}, \end{cases}$$

where $U_2^{\max}$ is the maximum utility when $p \in \Phi_2^{\varepsilon_2}$, $\varepsilon_2^{\max}$ is the maximum approximation ratio threshold acceptable to party 2, and $\varepsilon_2' \in [\varepsilon_2, \varepsilon_2^{\max}]$ indicates the ratio to which path $\mathbf{x}$ approximates a Pareto-optimal solution for party 2. The linear term ensures that the utility decreases

proportionally as p deviates from ε_2 within the range $[\varepsilon_2, \varepsilon_2^{\max}]$.

Definition 11 is essentially a variation of the ultimatum game. When two parties cannot reach a consensus, one of them will relax its conditions to seek cooperation, but will not exceed the bottom line. In this definition, we assume that party 2 will relax his conditions, and his benefits will also decrease when he relaxes his conditions. According to the payoff functions of both parties, when consensus is reached, both parties achieve the highest payoff, and the corresponding strategy is the Nash equilibrium.

Theorem 5. *Let $\varepsilon_1 > 0$, $\varepsilon_2 > 0$, $\varepsilon = \min\{\varepsilon_1, \varepsilon_2\}$, and $1+\varepsilon_1$ and $1+\varepsilon_2$ represent the approximation ratios for party 1 and 2, respectively. Let $\delta = \frac{n\log(nw^{\max})}{\log(1+\varepsilon)}$ and $k = \max\{k_1, k_2\}$ represent the maximum number of objectives among all parties. The expected number of generations of EMPMO$^{SP}_{simple}$ applied to BPMOSP, which achieves the Nash equilibrium of the ultimatum game for both parties is bounded by:*

$$O\left(n^3 \cdot \delta^{k-1} \log\left(n\delta^{k-1}\right)\right).$$

Proof. In the first phase of EMPMO$^{\text{SP}}_{\text{simple}}$, the Pareto approximate solution sets for both parties are independently searched. In Ref. [22], the entire process is divided into $n-1$ stages based on path length. In each stage, a path is selected for mutation until the population reaches the conditions for approximate dominance at that stage. The mutation strategy used is to add or delete nodes with equal probability only at the endpoints of the path. We removed two of the simplification operations and obtained the running time for each party is bounded by:

$$O\left(n^3 \cdot \delta_m^{k_m-1} \cdot \log\left(n\delta_m^{k_m-1}\right)\right),$$

where $\delta_m = \frac{n\log(nw_m^{\max})}{\log(1+\varepsilon_m)}$. Thus, the overall running time for the first phase is the sum of the running times for both parties:

$$\begin{aligned} &O\left(\sum_{m=1}^{2} n^3 \cdot \delta_m^{k_m-1} \log\left(n\delta_m^{k_m-1}\right)\right) \\ \leq &O\left(n^3 \cdot \delta^{k-1} \log\left(n\delta^{k-1}\right)\right), \end{aligned}$$

where $\delta = \frac{n\log(nw^{\max})}{\log(1+\varepsilon)}$. In the second phase, we sequentially check each of the first party's approximate Pareto solutions to verify whether it satisfies the Nash equilibrium of the ultimatum game as defined in Definition 11. This step does not require additional evaluation rounds. □

C. Runtime Analysis of Consensus-based Evolutionary Bi-party Multi-objective Optimizer

The EMPMO$^{\text{SP}}_{\text{simple}}$ algorithm can be extended to solve the BPMOSP with asymmetric influence between decision-makers, where one party is required to relax its predefined approximation ratio. To promote fairness, the evolutionary process aims to maximize consensus between the two parties, thereby facilitating a more balanced negotiation.

It is well established that when a path's weight is determined by a single value, the shortest path possesses the optimal substructure property [43], meaning that every subpath of the shortest path is itself a shortest path. Similarly, in the case of the common shortest path with multiple weights for two parties, analogous properties hold. Lemma 6 formalizes the optimal substructure property for the common shortest path.

Lemma 6. *Given a directed weighted graph $G = (V, E, W)$, if path $\mathbf{x} = (v_0, v_1, ..., v_{l-1}, v_l)$ is a common shortest path from source vertex v_0 to target vertex v_l, then $\mathbf{x}' = (v_0, v_1, ..., v_{l-1})$ is a common shortest path from source vertex v_0 to target vertex v_{l-1}.*

Proof. Let $\mathbf{x}$ be a common shortest path from source vertex v_0 to target vertex v_l. This implies that, among all other paths from v_0 to v_l, no path dominates $\mathbf{x}$ for all parties simultaneously. Let $F_m(\mathbf{x}) = (f_{m1}, \ldots, f_{mk_m})$ and $w_m(v_{l-1}, v_l) = (w_{m1}, \ldots, w_{mk_m})$. Then, we have $F_m(\mathbf{x}') = (f_{m1} - w_{m1}, \ldots, f_{mk_m} - w_{mk_m})$.

Assume, for contradiction, that $\mathbf{x}'$ is not a common shortest path from v_0 to v_{l-1}. This means there exists a path $\mathbf{z}' = (v_0, v_1, \ldots, v_{l-1})$ from v_0 to v_{l-1} that dominates $\mathbf{x}'$ for all parties. Let $F_m(\mathbf{z}') = (f'_{m1}, \ldots, f'_{mk_m})$. By Definition 2, we have $f'_{mk} \leq f_{mk} - w_{mk}$ for all $1 \leq k \leq k_m$, and $F_m(\mathbf{z}') \neq F_m(\mathbf{x}')$.

Using $\mathbf{z}'$, we can construct a new path $\mathbf{z} = (v_0, v_1, \ldots, v_{l-1}, v_l)$ with $F_m(\mathbf{z}) = (f'_{m1} + w_{m1}, \ldots, f'_{mk_m} + w_{mk_m})$. Since $f'_{mk} \leq f_{mk} - w_{mk}$ for all $1 \leq k \leq k_m$, it follows that $F_m(\mathbf{z}) \succeq F_m(\mathbf{x})$, which contradicts the assumption that $\mathbf{x}$ is a common shortest path. Therefore, the assumption is false, and $\mathbf{x}'$ must be a common shortest path from v_0 to v_{l-1}. □

Lemma 6 establishes that a common solution exists at each path length. Building on this result, if a common solution is found at every path length, then as the path length increases, a common solution to each target vertex can be identified. Therefore, we propose a consensus-based evolutionary multi-party multi-objective optimizer (EMPMO$^{\text{SP}}_{\text{cons}}$), as presented in Algorithm 6 for the BPMOSP. The details are as follows:

Algorithm 6 Consensus-based Evolutionary Multi-party Multi-objective Optimizer (EMPMO$^{SP}_{cons}$)

Require: Parameter r controlling the box size
1: Initialize the population $P \leftarrow \{(1)\}$;
2: **while** termination condition is not met **do**
3: Randomly select an individual $\mathbf{x} \in P$;
4: Apply the mutation operator on $\mathbf{x}$ to get $\mathbf{x}'$;
5: **if** $\exists m, \nexists \mathbf{z} \in P, F_m(\mathbf{z}) \succ F_m(\mathbf{x}') \vee b_r(F_m(\mathbf{z})) \succ b_r(F_m(\mathbf{x}'))$ **then**
6: $P \leftarrow (P \setminus \{\mathbf{z} \in P \mid \forall m', b_r(F_{m'}(\mathbf{x}')) \succeq b_r(F_{m'}(\mathbf{z}))\}) \cup \{\mathbf{x}'\}$;
7: **end if**
8: **end while**

During the initialization of the population, we set it to $\{(1)\}$. Drawing inspiration from common one-bit mutation, we define a novel mutation operator tailored for the BPMOSP, as detailed in Definition 12. This mutation operator randomly selects a vertex within the path to mutate. In each iteration, a solution

is randomly chosen for mutation. Subsequently, the population is updated. If no solution in the population dominates the mutated solution, and no box index corresponding to any party dominates that of the mutated solution, the new solution is added to the population. Subsequently, any solutions that are dominated by the new solution or share the same box index for all parties are removed from the population. The input parameter r in the algorithm represents the size of the box, and its value is set to $(1+\varepsilon)^{1/(n-1)}$ in subsequent runtime analysis and experiments.

Definition 12. *Each mutation operation randomly and uniformly performs one of the following two operations:*

***Add**: Randomly select a vertex v_i in the path $\mathbf{x} = (v_0, v_1, \ldots, v_{l-1}, v_l)$, where $0 \leq i \leq l$.*

- *If $v_i \neq v_l$ (i.e., v_i is not the target vertex of the path), and there exists $v' \in V$ such that $(v_i, v') \in E$ and $(v', v_{i+1}) \in E$, the new path after mutation is $\mathbf{x}' = (v_0, \ldots, v_i, v', v_{i+1}, \ldots, v_l)$.*
- *If $v_i = v_l$ (i.e., v_i is the target vertex of the path), and there exists $v' \in V$ such that $(v_l, v') \in E$, the new path after mutation is $\mathbf{x}' = (v_0, v_1, \ldots, v_l, v')$.*

***Delete**: Randomly select a vertex v_i in the path $\mathbf{x} = (v_0, v_1, \ldots, v_{l-1}, v_l)$, where $1 \leq i \leq l-1$.*

- *If $1 \leq i \leq l-2$ and $(v_i, v_{i+2}) \in E$, the new path after mutation is $\mathbf{x}' = (v_0, \ldots, v_i, v_{i+2}, \ldots, v_l)$.*
- *If $i = l-1$, the new path after mutation is $p' = (v_0, \ldots, v_{l-1})$.*

A more detailed explanation is provided in Example 2.

Example 2. *Using the weighted directed graph in Figure 2, the common solutions from vertex 1 to all other vertices can be determined as follows:*

- *From vertex 1 to vertex 2: $(1, 2)$,*
- *From vertex 1 to vertex 3: $(1, 3)$,*
- *From vertex 1 to vertex 4: $(1, 3, 4)$,*
- *From vertex 1 to vertex 5: $(1, 3, 4, 5)$.*

Let $\mathbf{x} = (1, 3, 4)$ and $\mathbf{z} = (1, 4)$.

TABLE II: All possible paths from source vertex 1 to target vertices except 5.

Paths	Objective Value for Party 1	Objective Value for Party 2
$(\mathbf{1}, \mathbf{2})$	$(\mathbf{1}, \mathbf{2})$	$(\mathbf{2}, \mathbf{4})$
$(1, 2, 3)$	$(4, 5)$	$(4, 5)$
$(\mathbf{1}, \mathbf{3})$	$(\mathbf{3}, \mathbf{2})$	$(\mathbf{3}, \mathbf{5})$
$(\mathbf{1}, \mathbf{3}, \mathbf{4})$	$(\mathbf{5}, \mathbf{3})$	$(\mathbf{4}, \mathbf{6})$
$(1, 2, 3, 4)$	$(6, 6)$	$(5, 6)$

During the mutation process, the path length increases incrementally. When the length is 1, non-common solutions like $\mathbf{z}$ are generated, but the common solution $\mathbf{x}' = (1, 3)$ is present in the population. Although a common solution from 1 to 4 is not immediately available, the population contains the common solution from 1 to 3. Through mutation, path $\mathbf{x}$ (a common solution from 1 to 4) can be identified.

According to the update rules defined in Definition 12, path $\mathbf{z}$ will naturally be removed from the population. Even if $\mathbf{x}$ is not obtained through mutation, the population will retain at least one solution satisfying conditions for each endpoint. If $\mathbf{z}$ is retained, it indicates that it satisfies the approximate domination criteria for both parties, allowing a solution that meets the conditions to still be found.

Thus, for any path length and target vertex, it suffices to identify at least one new common solution at each step.

To analyze the expected running time of $\text{EMPMO}_{\text{cons}}^{\text{SP}}$ on BPMOSP, several lemmas are required to assist in the proof.

Lemma 7. *Let $r > 1$ and $m \in \{1, 2\}$. When using $EMPMO_{cons}^{SP}$ to optimize all F_m, the maximum population size is upper bounded by*

$$\min_{1 \leq m \leq M} (n-1) \cdot (\lfloor \log_r((n-1)w_m^{\max}) \rfloor + 1)^{k_m - 1} + 1$$

Proof. To determine the maximum population size, we divide the analysis into two parts. The first part considers the optimization of any single F_m, while the second part extends the analysis to all F_m.

Let $\text{Path}_{1,i}$ represent all possible paths from source vertex 1 to target vertex i. For an objective value f_{mk}, there is a cardinality $|b_r(f_{mk}(\text{Path}_{1,i}))|$ for the corresponding m-th box. Given the following:

$$\begin{aligned} |b_r(f_{mk}(\text{Path}_{1,i}))| &\leq b_r(f_{mk}^{\max}) - b_r(f_{mk}^{\min}) + 1 \\ &= b_r((n-1)w_m^{\max}) - b_r(1) + 1 \\ &= \lfloor \log_r((n-1)\, w_m^{\max}) \rfloor + 1. \end{aligned}$$

Considering the population selecting rules in $\text{EMPMO}_{\text{cons}}^{\text{SP}}$, at most one candidate solution from each non-dominated box can enter the population. Therefore, we have:

$$\begin{aligned} |b_r(F_m(\text{Path}_{1,i}))| &\leq \prod_{k=2}^{k_m} |b_r(f_{mk}(\text{Path}_{1,i}))| \\ &\leq (\lfloor \log_r((n-1)w_m^{\max}) \rfloor + 1)^{k_m - 1} \end{aligned}$$

Thus, for the m-th party, $\text{Path}_{1,i}$ can have at most $(\lfloor \log_r((n-1)w_m^{\max}) \rfloor + 1)^{k_m - 1}$ distinct boxes that can enter the population. Excluding the case where the source vertex and the target vertex are the same, there are $n-1$ distinct target vertices. Therefore, for any F_m, there are at most $(n-1) \cdot (\lfloor \log_r((n-1)w_m^{\max}) \rfloor + 1)^{k_m - 1}$ boxes that can be included in the population.

In the second part, the condition for saving a solution is stricter than considering just one party alone. Hence, when optimizing all F_m, the maximum population size is:

$$\min_{1 \leq m \leq M} (n-1) \cdot (\lfloor \log_r((n-1)w_m^{\max}) \rfloor + 1)^{k_m - 1} + 1$$

The final constant 1 accounts for the case where the path length is zero, corresponding to the set $\{(1)\}$. □

Lemma 8. *Let $r > 1$, $m \in \{1, 2\}$, and $\mathbf{x} = (v_0, v_1, \ldots, v_{i-1}, v_i)$ be a path of length l, where $0 \leq i \leq n-2$. Let $\mathbf{z} = (v_0, u_1, \ldots, u_{j-1}, u_j) \in P$ be a path in the population P such that $u_j = v_i$. If $F_m(\mathbf{z}) \succeq_{r^i} F_m(\mathbf{x})$ for all m, then $F_m(\mathbf{z}') \succeq_{r^i} F_m(\mathbf{x}')$ for all m, where $\mathbf{x}' = (v_0, v_1, \ldots, v_i, v_{i+1})$ and $\mathbf{z}' = (v_0, u_1, \ldots, u_j, v_{i+1})$.*

Proof. Given that $F_m(\mathbf{z}) \succeq_{r^i} F_m(\mathbf{x})$, it follows that $f_{mk}(\mathbf{z}) \leq r^i \cdot f_{mk}(\mathbf{x})$ for all $1 \leq k \leq k_m$. For the extended paths $\mathbf{z}'$ and $\mathbf{x}'$, we have:

$$f_{mk}(\mathbf{z}') = f_{mk}(\mathbf{z}) + w_{mk}((u_j, v_{i+1}))$$
$$f_{mk}(\mathbf{x}') = f_{mk}(\mathbf{x}) + w_{mk}((v_i, v_{i+1})).$$

From $f_{mk}(\mathbf{z}) \leq r^i \cdot f_{mk}(\mathbf{x})$, it follows that:

$$\begin{aligned} f_{mk}(\mathbf{z}) + w_{mk}(v_i, v_{i+1}) &\leq r^i \cdot f_{mk}(\mathbf{x}) + w_{mk}(v_i, v_{i+1}) \\ &\leq r^i \cdot f_{mk}(\mathbf{x}) + r^i \cdot w_{mk}(v_i, v_{i+1}). \end{aligned}$$

Thus:

$$f_{mk}(\mathbf{z}') \leq r^i \cdot f_{mk}(\mathbf{x}'),$$

which implies that $F_m(\mathbf{z}') \succeq_{r^i} F_m(\mathbf{x}')$. □

Lemma 9. *Let $r > 1$, $m \in \{1, 2\}$, and $\mathbf{x} = (v_1, \ldots, v_i)$ be a path in the solution space. The set P_i consists of paths in the population P whose target vertices are v_i. If P_i satisfies $F_m(P_i) \succeq_{r^j} F_m(\mathbf{x})$ for all m, then the set P_i', consisting of paths with the same target vertex as $\mathbf{x}$ in the subsequent population P' obtained through mutation, satisfies $F_m(P_i') \succeq_{r^{j+1}} F_m(\mathbf{x})$ for all m.*

Proof. From the update rule of the population in Algorithm 6, for any $\mathbf{x} \in P_i \subset P$, there exists $\mathbf{z}' \in P_i' \subset P'$ such that $b_r(F_m(\mathbf{z}')) \succeq b_r(F_m(\mathbf{z}))$ for all m.

Given $b_r(F_m(\mathbf{z}')) \succeq b_r(F_m(\mathbf{z}))$ and $F_m(\mathbf{z}) \succeq_{r^j} F_m(\mathbf{x})$ (since $F_m(P_i) \succeq_{r^j} F_m(\mathbf{x})$ and $\mathbf{z} \in P_i$), it follows that:

$$\lfloor \log_r(f_{mk}(\mathbf{z}')) \rfloor \leq \lfloor \log_r(f_{mk}(\mathbf{z})) \rfloor,$$

then we have

$$\frac{\log(f_{mk}(\mathbf{z}'))}{\log r} - 1 \leq \frac{\log(f_{mk}(\mathbf{z}))}{\log r},$$

and then

$$\log(f_{mk}(\mathbf{z}')) \leq \log(r \cdot f_{mk}(\mathbf{z})).$$

Thus,

$$f_{mk}(\mathbf{z}') \leq r \cdot f_{mk}(\mathbf{z}).$$

By induction, since $f_{mk}(\mathbf{z}) \leq r^j f_{mk}(\mathbf{x})$, it follows that:

$$f_{mk}(\mathbf{z}') \leq r \cdot r^j f_{mk}(\mathbf{x}) = r^{j+1} f_{mk}(\mathbf{x}).$$

This demonstrates that $F_m(P_i') \succeq_{r^{j+1}} F_m(\mathbf{x})$ for all m. □

Theorem 6. *Let $\varepsilon > 0$, $m \in \{1, 2\}$, $\delta = \frac{n \log(nw^{\max})}{\log(1+\varepsilon)}$ and k represent the maximum number of objectives among all parties. When EMPMO$^{SP}_{cons}$ is applied to BPMOSP, it achieves a $(1+\varepsilon)$-approximation with an expected number of generations bounded by:*

$$O\left(n^4 \cdot \delta^{k-1} \cdot \log\left(n\delta^{k-1}\right)\right).$$

Proof. Since there are n vertices in the graph, the maximum path length is $n-1$. Based on the path length, we divide the entire evolution process into $n-1$ stages. At the end of the i-th stage, we have $F_m(P) \succeq_{r^{i+1}} F_m(S_{i+1})$ for all m, where P represents the population, $0 \leq i \leq n-2$ is the path length, S_{i+1} represents all possible paths in the solution space with path lengths not exceeding $i+1$, and $r = (1+\varepsilon)^{\frac{1}{n-1}}$. After completing all $n-1$ stages, it holds that $r^{n-1} = 1 + \varepsilon$.

At the initialization of the population, let $P = \{(v_1)\} = S_0$, which satisfies $F_m(P) \succeq_{r^0} F_m(S_0)$ for all m. At the beginning of the i-th stage, it holds that $F_m(P) \succeq_{r^i} F_m(S_i)$ for all m.

For any $\mathbf{x} \in S_i$, according to Lemma 8, there exists a path $\mathbf{z}$ in the population with the same target vertex as $\mathbf{x}$ such that $F_m(\mathbf{z}') \succeq_{r^i} F_m(\mathbf{x}')$ for all m, where $\mathbf{x}' \in S_{i+1}$ and $\mathbf{z}'$ is a mutation of $\mathbf{z}$. By Lemma 9, the subsequent population P' satisfies $F_m(P') \succeq_{r^{i+1}} F_m(\mathbf{x}')$ for all m. The i-th stage ends when $F_m(P') \succeq_{r^{i+1}} F_m(S_{i+1})$ is satisfied for all m.

According to Lemma 6, we keep the form of $F_m(\mathbf{x})$, $w_m(v_{l-1}, v_l)$ and $F_m(\mathbf{x}')$ unchanged and let $F_m(\mathbf{z}) = (f'_{m1}, \ldots, f'_{mk_m})$ and $F_m(\mathbf{z}') = (f'_{m1} - w_{m1}, \ldots, f'_{mk_m} - w_{mk_m})$. If $\mathbf{x}'$ is a approximate common shortest path from v_0 to v_{l-1} but $\mathbf{z}'$ is not, we have $F_m(\mathbf{x}') \succeq_{r^j} F_m(\mathbf{z}')$,that is $(f_{mk} - w_{mk}) \leq r^j(f'_{mk} - w_{mk})$ for all $1 \leq k \leq k_m$. After scaling the inequality, we can get $f_{mk} \leq r^j f'_{mk} - (r^j - 1)w_{mk} \leq r^{j+1} f'_{mk} - (r^j - 1)w_{mk} \leq r^{j+1} f'_{mk}$ because of $r > 1$, that is $F_m(\mathbf{x}) \succeq_{r^{j+1}} F_m(\mathbf{z})$. Therefore, we know that a common solution can be found in the first stage, and an approximate common solution of the corresponding length can be found through mutation in the i-th stage. If this solution is deleted during population evolution, it means that there are other approximate common solutions that are closer to the true common solution than it.

For the mutation operation, the probability of selecting a solution from the population is $\frac{1}{|P|}$, the probability of performing the Add operation in the mutation operator on this solution is $\frac{1}{2}$, and the probability of selecting the endpoint of the path and adding an edge is at least $\frac{1}{n} \cdot \frac{1}{n-1}$. Thus, the probability of successfully executing the operation in Lemma 8 is at least $\frac{1}{2 \cdot |P| \cdot n(n-1)} \geq \frac{1}{2n(n-1)P_{\max}}$. According to Lemma 7, $P_{\max} = \min_{1 \leq m \leq M}(n-1) \cdot (\lfloor \log_r((n-1)w_m^{\max}) \rfloor + 1)^{k_m - 1} + 1$.

Hence, the maximum expected number of generations for the i-th stage is

$$2n(n-1) \cdot P_{\max} \cdot \sum_{j=1}^{P_{\max}} \frac{1}{j} = O\left(n^2 \cdot P_{\max} \cdot \log P_{\max}\right).$$

Let $k = \max\{k_1, \ldots, k_M\}$. Since $r = (1+\varepsilon)^{\frac{1}{n-1}}$, $w_m^{\max} \leq w^{\max}$, we have

$$\begin{aligned} &O\left(n^2 \cdot P_{\max} \cdot \log P_{\max}\right) \\ =&O\left(n^2 \cdot n\left(\log_r(nw^{\max})\right)^{k-1} \log\left(n \cdot (\log_r(nw^{\max}))^{k-1}\right)\right) \\ =&O\left(n^3 \left(\frac{n\log(nw^{\max})}{\log(1+\varepsilon)}\right)^{k-1} \log\left(n\left(\frac{n\log(nw^{\max})}{\log(1+\varepsilon)}\right)^{k-1}\right)\right) \\ =&O\left(n^3 \cdot \delta^{k-1} \cdot \log\left(n\delta^{k-1}\right)\right). \end{aligned}$$

Since there are $n-1$ stages in total, the overall expected number of generations is at most

$$O\left(n^4 \cdot \delta^{k-1} \cdot \log\left(n\delta^{k-1}\right)\right).$$

□

D. Comparison Between Bi-party Multi-objective Shortest Path and Multi-objective Shortest Path

Consider the case where $k_1 = k_2 = 2$, representing a bi-party bi-objective shortest path (SP) problem. Compared to the

generalized four-objective SP problem, the time complexities are as follows.

Bi-objective SP Problems: As stated in [22], the upper bound on the expected number of generations is:

$$O\left(n^3 \cdot \frac{n\log(nw^{\max})}{\log(1+\varepsilon)} \cdot \log\left(n \cdot \frac{n\log(nw^{\max})}{\log(1+\varepsilon)}\right)\right).$$

This bound also applies to $\text{EMPMO}^{\text{SP}}_{\text{simple}}$ for the bi-party bi-objective SP problem, but with a larger approximation ratio.

Four-objective Problems: The upper bound on the expected number of generations is:

$$O\left(n^3 \cdot \left(\frac{n\log(nw^{\max})}{\log(1+\varepsilon)}\right)^3 \cdot \log\left(n \cdot \left(\frac{n\log(nw^{\max})}{\log(1+\varepsilon)}\right)^3\right)\right).$$

Bi-party Bi-objective SP Problem: Using $\text{EMPMO}^{\text{SP}}_{\text{cons}}$, the upper bound on the expected number of generations is:

$$O\left(n^4 \cdot \frac{n\log(nw^{\max})}{\log(1+\varepsilon)} \cdot \log\left(n \cdot \frac{n\log(nw^{\max})}{\log(1+\varepsilon)}\right)\right).$$

In all cases, the approximation accuracy ε is consistent across the bi-objective, four-objective, and bi-party bi-objective problems. While $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ modifies the mutation operator, its running time remains lower than that for the four-objective problem and slightly higher than that for the bi-objective problem. Importantly, unlike $\text{EMPMO}^{\text{SP}}_{\text{simple}}$, $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ does not relax the approximation accuracy, thereby yielding higher-quality solutions.

V. Experiments

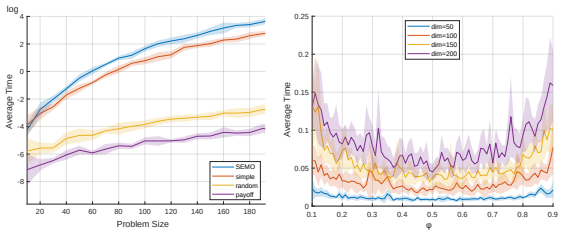
In this section, we conduct short experiments on the artificial problem BPAOAZ and the bi-party multi-objective UAV path planning (BPUAVPP) to complement the theoretical results.

A. On Artificial Problem BPAOAZ

In this subsection, we analyze the average running times of the MOEA, i.e., SEMO, and three MPMOEAs: $\text{EMPMO}_{\text{simple}}$, $\text{EMPMO}_{\text{random}}$, and $\text{EMPMO}_{\text{payoff}}$, on the artificial problem BPAOAZ. Each algorithm was independently executed 10 times. The results, illustrated in Fig. 3a, provide a comparative evaluation of their performance.

SEMO is designed to identify the complete Pareto set of the associated multi-objective optimization problem, whereas the other three algorithms focus on finding the common Pareto set. Among these, $\text{EMPMO}_{\text{payoff}}$ serves as a theoretical performance baseline for comparison. The results demonstrate that SEMO exhibits the longest runtime, followed by $\text{EMPMO}_{\text{simple}}$, $\text{EMPMO}_{\text{random}}$, and $\text{EMPMO}_{\text{payoff}}$, in descending order of execution time. These experimental findings corroborate the theoretical results presented in Section III.

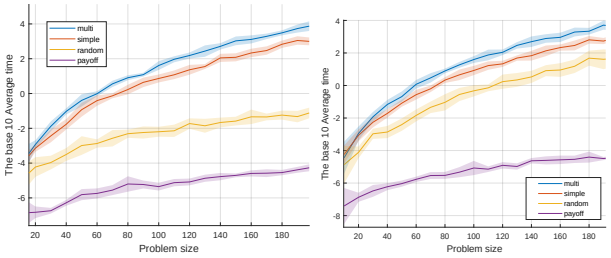
It is noteworthy that Theorem 2 establishes that the expected runtime of $\text{EMPMO}_{\text{random}}$ on BPAOAZ is bounded by $O\left(\left(\frac{1}{\varphi}+\frac{1}{1-\varphi}\right)\frac{n^2}{2}\log n\right)$, consistent with the runtime complexity of $\text{EMPMO}_{\text{simple}}$. However, as shown in Fig. 3a, when the parameter $\varphi = 0.5$ is set, the actual runtime of $\text{EMPMO}_{\text{random}}$ is closer to the theoretical lower bound $\Omega(n\log n)$ and is comparable to that of $\text{EMPMO}_{\text{payoff}}$.



(a) Logarithmic variation of running time with problem size (b) Variation of $\text{EMPMO}_{\text{random}}$ running time with φ

Fig. 3: The average runtime of SEMO, $\text{EMPMO}_{\text{simple}}$, $\text{EMPMO}_{\text{random}}$, and $\text{EMPMO}_{\text{payoff}}$ on artificial problem BPAOAZ and the y-axis is the average runtime in base 10.

This discrepancy arises from the significant influence of the selection probability φ on runtime. Fig. 3b further illustrates the runtime of $\text{EMPMO}_{\text{random}}$ as a function of φ across different problem sizes. For BPAOAZ, which features symmetric subproblems for both parties, $\varphi = 0.5$ is empirically identified as the optimal parameter. Additionally, as the problem size increases, the variability in runtime becomes more pronounced, peaking at the extremes of the parameter range. Due to the added randomness in the selection process, $\text{EMPMO}_{\text{random}}$ exhibits greater variance compared to $\text{EMPMO}_{\text{simple}}$. This is evident from the broader shaded regions in Fig. 3b, which represent the range of runtime fluctuations. In addition, for a more rigorous proof, we add two experiments. One experiment (Fig. 4a) shows the comparison results when $\varphi = 0.95$ and further proves that the value of φ will affect the running time of $\text{EMPMO}_{\text{random}}$ to find the common solution. The other experiment (Fig. 4b) simulates the derivation process of $\text{EMPMO}_{\text{random}}$ and proves the correctness of the lower bound of the running time.

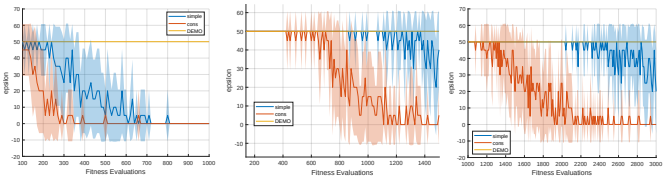


(a) The running time of $\text{EMPMO}_{\text{random}}$ with $\varphi = 0.95$ (b) Variation of the running time of $\text{EMPMO}_{\text{random}}$

Fig. 4: The average runtime of SEMO, $\text{EMPMO}_{\text{simple}}$, $\text{EMPMO}_{\text{random}}$, and $\text{EMPMO}_{\text{payoff}}$ on artificial problem BPAOAZ and the y-axis is the average runtime in base 10.

B. On Bi-party Multi-objective UAV Path Planning

In this experiment, we evaluate the performance of DEMO, $\text{EMPMO}^{\text{SP}}_{\text{simple}}$, and $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ on the BPMOSP problem, which is based on the bi-party multi-objective UAV path planning (BPUAVPP) problem proposed by Chen et al. [2]. The problem involves two stakeholders: the efficiency party and the safety party. To account for the computational complexity of



(a) UAV1 (b) UAV2 (c) UAV3

Fig. 5: The largest minimum approximate degree of DEMO, $\text{EMPMO}^{\text{SP}}_{\text{simple}}$, and $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ on BPUAVPP.

the underlying optimizer, we simplify the BPMOSP problem into a combinatorial optimization problem that seeks the shortest path on a finite set of nodes, disregarding continuous coordinates. Additionally, all constraints in the problem are omitted. The problem construction ensures the existence of a common solution.

- The efficiency party focuses on minimizing the path length and the mission hover point distance, formulated as follows:

$$f_{\text{length}} = \sum_{i=0}^{n-1} ||\mathbf{g}_i||, \quad f_{\text{distance}} = \sum_{k=0}^{K-1} \min_i ||\mathbf{p}_i - \mathbf{p}_k^{\text{job}}||,$$

  where g_i represents the length of the i-th traversed edge, p_k^{job} denotes the k-th preset UAV hover point, and p_i is the i-th discrete trajectory point.
- The safety party aims to minimize the risks to pedestrians and property, as described by the following objectives:

$$f_{\text{fatal}} = \sum_{i=0}^{n} c_{r_p}(x_i, y_i, z_i), \quad f_{\text{eco}} = \sum_{i=0}^{n} \psi(z_i),$$

  where $c_{r_p}(x, y, z) = P_{\text{crash}} S_h \sigma_p(x, y) R_f^P(z)$ represents the fatality risk cost, P_{crash} denotes the crash probability, which depends on UAV hardware and software reliability, S_h represents the crash impact area, $\sigma_p(x, y)$ indicates population density at location (x, y), and $R_f^P(z)$ correlating with the kinetic energy of the impact and obscuration factors, and ψ is a lognormal distribution function of the flight height z_i.

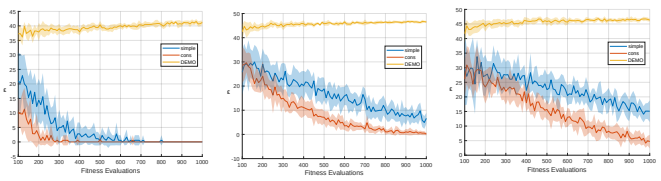
The experiments included three test cases for the BPUAVPP problem, consisting of 10, 30, and 50 vertices, referred to as UAV1, UAV2, and UAV3, respectively. Figures 5a - 5c depict the worst performance trends of the three algorithms under comparison across these test cases. The x-axis represents the number of evaluations, while the y-axis denotes the largest minimum degree to which all solutions in the population can approximately dominate the common solution for the same endpoint, which can be expressed as: $\max_{1 \le i \le |P|} \varepsilon_{x_i}$, where P represents the population, and $x_i = (v_0, ..., u_j)$ is the i-th solution in P. Let Φ_{u_j} represent the set of solutions in the common solution set whose endpoint is u_j, then ε_{x_i} represents the minimum degree to which x_i can approximately dominate all solutions in Φ_{u_j}. In the experiment, the value of ε_{x_i} is calculated using the bisection method. This metric reflects the population's worst approximation quality relative to the common solution, with smaller values indicating better population quality.

The results clearly show that for the multi-objective optimization algorithm, the maximum ε-value did not converge. This is because the algorithm aims to identify the complete Pareto set (PS) for the multi-objective version of BPUAVPP, including non-common solutions. In contrast, the worst approximations of the other two algorithms converge as the number of evaluations increases, indicating that they are able to find common solutions over time. Among them, our proposed $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ is able to find common solutions faster.

In the UAV1 case, a unique common solution can be identified. Since UAV1 consists of only 10 nodes, both the simple and cons algorithms eventually find the true common solution after sufficient evaluations, achieving $\varepsilon = 0$. A comparison across different problem complexities reveals that as the problem becomes more challenging, the approximation quality decreases for the same number of evaluations.

These experimental results strongly support the theoretical analysis presented earlier. First, the BPUAVPP problem cannot be effectively solved using multi-objective optimization algorithms, as the population introduces numerous non-common solutions. Second, $\text{EMPMO}^{\text{SP}}_{\text{simple}}$, which only enforces agreement between parties in the final stage, requires the algorithm to explore the complete PS during the evolutionary phase to achieve higher precision at the end. This slows the convergence speed. Finally, $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ not only identifies the correct common solution, as shown in the UAV1 and UAV2 cases, but also achieves higher precision compared to $\text{EMPMO}^{\text{SP}}_{\text{simple}}$.

We also conducted experiments on the average minimum degree to which all solutions in the population can approximately dominate the common solution for the same endpoint, and the results strongly supported the previous theoretical analysis. Figures 6a - 6c depict the performance trends of the three algorithms under comparison across these test cases. The x-axis represents the number of evaluations, while the y-axis denotes the average minimum degree to which all solutions in the population can approximately dominate the common solution for the same endpoint. This metric reflects the population's overall approximation quality relative to the common solution, with smaller values indicating better population quality.



(a) UAV1 (b) UAV2 (c) UAV3

Fig. 6: The average minimum approximate degree of DEMO, $\text{EMPMO}^{\text{SP}}_{\text{simple}}$, and $\text{EMPMO}^{\text{SP}}_{\text{cons}}$ on BPUAVPP.

The results in Figures 6a - 6c clearly demonstrate that for multi-objective optimization algorithms, the ε-value shows a slight upward trend as the number of evaluations increases. This is because the algorithm aims to identify the complete

Pareto set (PS) for the multi-objective version of BPUAVPP, including non-common solutions. As evaluations increase, the inclusion of more non-common solutions leads to a rise in the population's overall ε-value. In contrast, the ε-values for the other two algorithms decrease over time.

VI. Conclusion

In this paper, we present the first mathematical analysis of the runtime of evolutionary algorithms applied to two-party multi-objective optimization problems. We demonstrate that multi-objective optimization algorithms are not suitable for bi-party multi-objective optimization problems, both in terms of runtime and the solution set. We then consider a transition from multi-objective optimization to multi-party multi-objective optimization by decoupling MPMOPs into two MOPs and separately optimizing them to compute their intersection for the common Pareto set. This approach is limited by the NP-hard nature of the problem, making it difficult to obtain exact solutions in polynomial time, and the final solution set composed of approximations often lacks an intersection. Finally, we propose two general bi-party multi-objective optimization frameworks, along with a customized algorithm for the bi-party multi-objective shortest path optimization. We provide theoretical guarantees for an approach that maintains a single population searching for a common solution, and demonstrate that considering the interaction between the two parties in each iteration optimizes both time efficiency and solution set quality. This paper lays the foundation for evolutionary multi-party multi-objective optimization analysis, which will be further extended in the future to analyze bi-party multi-objective optimization problems without a common solution , including the definition of optimality in the case of no common solution, the design of indicators, and specific proof analysis. In addition, we will also explore the population size in future work.

References

[1] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[2] K. Chen, W. Luo, X. Lin, Z. Song, and Y. Chang, "Evolutionary biparty multiobjective uav path planning: Problems and empirical comparisons," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 8, no. 3, pp. 2433–2445, 2024.

[3] C. Zhang, G. Wang, Y. Peng, G. Tang, and G. Liang, "A negotiation-based multi-objective, multi-party decision-making model for inter-basin water transfer scheme optimization," *Water Resources Management*, vol. 26, pp. 4029–4038, 2012.

[4] Z. Song, W. Luo, X. Lin, Z. She, and Q. Zhang, "On multiobjective knapsack problems with multiple decision makers," in *2022 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2022, pp. 156–163.

[5] W. Liu, W. Luo, X. Lin, M. Li, and S. Yang, "Evolutionary approach to multiparty multiobjective optimization problems with common pareto optimal solutions," in *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2020, pp. 1–9.

[6] Y. Chang, W. Luo, X. Lin, Z. She, and Y. Shi, "Multiparty multiobjective optimization by moea/d," in *2022 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, 2022, pp. 01–08.

[7] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[8] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength pareto evolutionary algorithm," *TIK report*, vol. 103, 2001.

[9] Q. Lin, J. Chen, Z.-H. Zhan, W.-N. Chen, C. A. C. Coello, Y. Yin, C.-M. Lin, and J. Zhang, "A hybrid evolutionary immune algorithm for multiobjective optimization problems," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 711–729, 2015.

[10] Z. She, W. Luo, Y. Chang, X. Lin, and Y. Tan, "A new evolutionary approach to multiparty multiobjective optimization," in *International Conference on Swarm Intelligence*. Springer, 2021, pp. 58–69.

[11] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[12] Y. Chang, W. Luo, X. Lin, Z. Song, and C. A. C. Coello, "Biparty multiobjective optimal power flow: The problem definition and an evolutionary approach," *Applied Soft Computing*, vol. 146, p. 110688, 2023.

[13] Z. Song, W. Luo, P. Xu, Z. Ye, and K. Chen, "An indicator based evolutionary algorithm for multiparty multiobjective knapsack problems," in *International Conference on Intelligent Information Processing*. Springer, 2024, pp. 233–246.

[14] N. Beume, B. Naujoks, and M. Emmerich, "SMS-EMOA: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[15] C. Qian, D.-X. Liu, and Z.-H. Zhou, "Result diversification by multi-objective evolutionary algorithms with theoretical guarantees," *Artificial Intelligence*, vol. 309, p. 103737, 2022.

[16] W. Zheng and B. Doerr, "Mathematical runtime analysis for the non-dominated sorting genetic algorithm II (NSGA-II)," *Artificial Intelligence*, vol. 325, p. 104016, 2023.

[17] ——, "Runtime analysis for the NSGA-II: proving, quantifying, and explaining the inefficiency for many objectives," *IEEE Transactions on Evolutionary Computation*, 2023.

[18] A. Opris, D.-C. Dang, F. Neumann, and D. Sudholt, "Runtime analyses of NSGA-III on many-objective problems," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2024, pp. 1596–1604.

[19] S. Ren, C. Bian, M. Li, and C. Qian, "A first running time analysis of the strength pareto evolutionary algorithm 2 (SPEA2)," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2024, pp. 295–312.

[20] M. Laumanns, L. Thiele, and E. Zitzler, "Running time analysis of multiobjective evolutionary algorithms on pseudo-boolean functions," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 2, pp. 170–182, 2004.

[21] C. Qian, Y. Yu, and Z.-H. Zhou, "An analysis on recombination in multi-objective evolutionary optimization," *Artificial Intelligence*, vol. 204, pp. 99–119, 2013.

[22] C. Horoba, "Exploring the runtime of an evolutionary algorithm for the multi-objective shortest path problem," *Evolutionary Computation*, vol. 18, no. 3, pp. 357–381, 2010.

[23] J. R. Koza, M. A. Keane, and M. J. Streeter, "What's AI done for me lately? genetic programming's human-competitive results," *IEEE Intelligent Systems*, vol. 18, no. 3, pp. 25–31, 2003.

[24] E. Benkhelifa, G. Dragffy, A. G. Pipe, and M. Nibouche, "Design innovation for real world applications, using evolutionary algorithms," in *2009 IEEE Congress on Evolutionary Computation*. IEEE, 2009, pp. 918–924.

[25] Y. Yu, X. Yao, and Z.-H. Zhou, "On the approximation ability of evolutionary optimization with application to minimum set cover," *Artificial Intelligence*, vol. 180, pp. 20–33, 2012.

[26] T. Friedrich, N. Hebbinghaus, F. Neumann, J. He, and C. Witt, "Approximating covering problems by randomized search heuristics using multi-objective models," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, 2007, pp. 797–804.

[27] O. Giel and P. K. Lehre, "On the effect of populations in evolutionary multi-objective optimization," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, 2006, pp. 651–658.

[28] C. Qian, Y. Yu, and Z.-H. Zhou, "An analysis on recombination in multi-objective evolutionary optimization," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, 2011, pp. 2051–2058.

[29] R. Allmendinger, J. Handl, and J. Knowles, "Multiobjective optimization: When objectives exhibit non-uniform latencies," *European Journal of Operational Research*, vol. 243, no. 2, pp. 497–513, 2015.

[30] M. A. Potter and K. A. De Jong, "A cooperative coevolutionary approach to function optimization," in *International conference on parallel problem solving from nature*. Springer, 1994, pp. 249–257.

[31] S. G. Ficici and J. B. Pollack, "A game-theoretic approach to the simple coevolutionary algorithm," in *International Conference on Parallel Problem Solving from Nature*. Springer, 2000, pp. 467–476.

[32] O. Giel and P. K. Lehre, "On the effect of populations in evolutionary multi-objective optimisation." *Evolutionary Computation*, vol. 18, no. 3, 2010.

[33] J. Xu, D. Pushp, K. Yin, and L. Liu, "Decision-making among bounded rational agents," in *International Symposium on Distributed Autonomous Robotic Systems*. Springer, 2022, pp. 273–285.

[34] J. G. Falcón-Cardona and C. A. C. Coello, "Indicator-based multi-objective evolutionary algorithms: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 53, no. 2, pp. 1–35, 2020.

[35] P. Xu, W. Luo, X. Lin, Y. Chang, and K. Tang, "Difficulty and contribution-based cooperative coevolution for large-scale optimization," *IEEE Transactions on Evolutionary Computation*, vol. 27, no. 5, pp. 1355–1369, 2022.

[36] P. Serafini, "Some considerations about computational complexity for multi objective combinatorial problems," in *Recent Advances and Historical Development of Vector Optimization*. Springer, 1987, pp. 222–232.

[37] S. Zajac and S. Huber, "Objectives and methods in multi-objective routing problems: a survey and classification scheme," *European Journal of Operational Research*, vol. 290, no. 1, pp. 1–25, 2021.

[38] D. Z. Chen, F. Trevizan, and S. Thiébaux, "Heuristic search for multi-objective probabilistic planning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 10, 2023, pp. 11 945–11 954.

[39] C. Horoba, "Analysis of a simple evolutionary algorithm for the multiobjective shortest path problem," in *Proceedings of the tenth ACM SIGEVO workshop on Foundations of genetic algorithms*, 2009, pp. 113–120.

[40] F. Bökler, "The multiobjective shortest path problem is NP-hard, or is it?" in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2017, pp. 77–87.

[41] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary computation*, vol. 10, no. 3, pp. 263–282, 2002.

[42] M. A. Nowak, K. M. Page, and K. Sigmund, "Fairness versus reason in the ultimatum game," *Science*, vol. 289, no. 5485, pp. 1773–1775, 2000.

[43] R. Bellman, "On a routing problem," *Quarterly of applied mathematics*, vol. 16, no. 1, pp. 87–90, 1958.