Subset Selection for Evolutionary Multi-Objective Optimization

Yu-Ran Gu, Chao Bian, Miqing Li, *Member, IEEE* and Chao Qian, *Senior Member, IEEE*

***Abstract*—Subset selection, which selects a subset of solutions according to certain criterion/indicator, is a topic closely related to evolutionary multi-objective optimization (EMO). The critical component of a multi-objective evolutionary algorithm (MOEA), environmental selection, is essentially a subset selection problem, i.e., selecting N solutions as the next-generation population from usually $2N$ candidates (where N denotes the size of the population). Another use of subset selection is the solution preprocessing procedure for decision-making, in which typically a few representatives are selected from the final population or a large-size archive, in order not to overwhelm the decision maker. Existing work for subset selection in EMO is focused on developing greedy algorithms, but may suffer from being trapped in local optima. In this paper, we approach the problem by providing a multi-objective evolutionary algorithmic framework. We consider several popular quality indicators for subset evaluation and present accelerated variants of a well-studied MOEA in the theoretical study area, Global Simple Evolutionary Multi-objective Optimizer (GSEMO), for each indicator. We conduct rigorous theoretical analyses of the acceleration procedure. Moreover, we prove that our algorithms can achieve the best-so-far approximation guarantee by the submodularity of the indicators. We finally empirically show the effectiveness and scalability of the proposed algorithms, in addition to the potentials to be further improved by introducing popular MOEAs (e.g., using NSGA-II and MOEA/D to replace GSEMO).**

***Index Terms*—Subset selection, Evolutionary multi-objective optimization, Quality indicators, Submodularity, multi-objective evolutionary optimizer (MOEAs), global simple evolutionary multi-objective algorithm (GSEMO) .**

I. Introduction

NOWADAYS, many real-world optimization problems require optimizing multiple objectives simultaneously. Multi-objective optimization has become a research hotspot in many areas, such as evolutionary computation [1] and machine learning [2]. Different from optimization problems with one objective function, multi-objective optimization problems use the Pareto-dominance relation to deal with the tradeoff between different objectives since these objectives are typically conflicting to each other.

In contrast to conventional algorithms which often obtain a single solution after one execution, multi-objective evolutionary algorithms (MOEAs) are able to find many Pareto optimal solutions approximating the whole Pareto front for multi-objective optimization problems, and thus offer decision makers with different choices. This leads to the emergence of many widely-used MOEAs over the last two decades such as NSGA-II/III [3, 4], MOEA/D [5] and SMS-EMOA [6].

Since the number of Pareto optimal solutions found by MOEAs can be far beyond which human beings can process, the problem of subset selection is an important topic in evolutionary multi-objective optimization (EMO). For example, the environmental selection operation in the evolutionary process [7], which selects some best solutions to form the next-generation population, is effectively a subset selection operation. Another noticeable example is in the decision making process, typically one would like to select a few (e.g., 5 to 10) most representative solutions (to present to the decision maker) out of the final population or even an unbounded archive [8]; a good example of such studies is [9].

In this paper, we study the subset selection problem in EMO, i.e., $\arg\max_{S\subseteq G} f(S)\ s.t.\ |S| \leq k$, where G is the set of elements $\boldsymbol{s}$ where $\boldsymbol{s} \in \mathbb{R}^m$ and f is the quality indicator function for subset evaluation. More concretely, f evaluates the goodness of a set, i.e., the quality of approximation to the Pareto front, typically measured by a quality indicator [10]. There are a range of quality indicators that can be used in subset selection (e.g., distance-based and volume-based ones) [10]. Among them, hypervolume [11], IGD [12], IGD+ [13] and R2 [14] are representative indicators.

Specifically, given a set of solutions and an indicator (which can reflect the quality of a solution set), the problem is to select a subset of the solution set that has the maximum quality on that indicator. This general subset selection can be directly applied to finding a representative set of the Pareto front approximation (e.g., obtained by an MOEA) under the assumption that the decision maker's preferences are aligned with the indicator. Another application of such subset selection to EMO lies in environmental selection of indicator-based MOEAs, where a new population is formed by selecting solutions from the old population and newly-generated solution(s), in order to maximize the quality of the population on that indicator.

Previous studies have shown that this problem with a submodular objective function is generally NP-hard [15], which means it is impossible for any algorithm to find an optimal solution efficiently. So, it is desirable for researchers to design algorithms locating solutions to this problem with a polynomial-time approximation guarantee. Along this line, the two most popular kinds of algorithms for solving this problem are the greedy and evolutionary algorithms.

For the problem we are tackling, Chen et al. [16] proposed a lazy greedy algorithm which exploits the properties of

Y.-R. Gu, C. Bian and C. Qian are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China (e-mails: {guyr, bianc, qianc}@lamda.nju.edu.cn). Chao Qian is the corresponding author.

M. Li is with School of Computer Science, University of Birmingham, Birmingham B15 2TT, U.K. (e-mail: m.li.8@bham.ac.uk)

monotonicity and submodularity of the hypervolume indicator. This greedy algorithm, which was inspired by the idea of the lazy evaluation [17], lazily recomputes the hypervolume contribution, i.e., the marginal gain of hypervolume indicator with respect to an element. According to the seminal work [18], the greedy algorithm achieves the state-of-the-art $(1-1/e)$-approximation guarantee when the quality indicator function is monotone and submodular.

More recently, Chen et al. [19] extended the conclusion in [16] with respect to IGD and IGD+ indicators and accelerated the calculation of these two indicators by using an array to store intermediate variables, thus avoiding unnecessary computation. Besides, Shang et al. [20] presented a greedy algorithm utilizing a hypervolume contribution approximation method as well as a benchmark test suite for large-scale subset selection problem in EMO [7].

Yet, greedy algorithms may fall into local optima during the execution due to their greedy nature, which was demonstrated using simple examples in [21]. Lately, some researches demonstrated promising performance of MOEAs on the subset selection problem and proved theoretical guarantees for these algorithms. In particular, Qian et al. [22] proved that the POSS algorithm (i.e., a variant of GSEMO [23]) can achieve the same $(1-1/e)$-approximation guarantee as the greedy algorithm, while having better performance in practice. Inspired by this, we propose a multi-objective evolutionary algorithmic framework to solve the subset selection problem in EMO. Generally speaking, we firstly formulate the problem into an optimization problem with two objectives, the quality of the subset and the size of the subset. Then we solve this bi-objective problem using an MOEA and select the optimal feasible solution of a subset selection problem in the final population. In fact, any MOEA can be utilized to solve this bi-objective problem.

With the aim of exploring our algorithmic framework, we design simple MOEAs based on GSEMO with regard to representative indicators (e.g., hypervolume, IGD, IGD+ and R2 indicators) and prove their theoretical guarantee. We accelerate the calculation of the indicators by developing an efficient method, which is applicable to any MOEA. Empirically, we show the superiority of the proposed framework. It is worth mentioning that there are some discussions on the hypervolume subset selection problem [24], whereas we study the general indicator-based subset selection problem.

The main contributions of this work can be summarized as follows.

- We propose a multi-objective evolutionary algorithmic framework for the subset selection problem in EMO. On top of it, we design accelerated variants of GSEMO in this framework with regard to representative indicators, and rigorously prove the acceleration ratio, which is $|R|$ for IGD and IGD+ and $|W|$ for R2, where R and W denote the reference set and the set of weight vectors, respectively.
- For the representative indicators, i.e., hypervolume, IGD, IGD+ and R2, we prove that our algorithms in the proposed framework can achieve the best-so-far approximation guarantee of $(1-1/e)$. The proof is based on the monotonicity and submodularity of these indicators. Note that these properties of hypervolume, IGD and IGD+ were proved previously, see in [19, 25], except for R2, which is proved in this paper.
- We carry out extensive experiments (i.e., on a range of test problems and different parameters and comparing with different algorithms) and demonstrate the superior performance of our algorithmic framework. Besides, we reveal that popular MOEAs (i.e., NSGA-II and MOEA/D) behave well on this problem, despite probably being limited by their nature.

The rest of this paper is organized as follows. Section II gives the preliminaries of this work. Section III presents our proposed algorithmic framework and specific algorithms for each representative indicator and proves the low computational complexity of these algorithms theoretically. In Section IV, we prove the submodularity of quality indicators. In view of this, we further give a brief theoretical analysis about the approximation guarantee to our algorithms. In Section V, we carry out a series of experiments to verify the performance. Finally, we conclude this work in Section VI.

II. Preliminaries

A. Multi-objective Optimization Problems

Definition 1 (Multi-objective Minimization Problem)**.** Given the feasible set $\mathcal{X}$ of solutions $\boldsymbol{s}$ and m objective functions $h_1,\ldots,h_m$, the corresponding multi-objective minimization problem is to find $\arg\min_{\boldsymbol{s}\in\mathcal{X}}(h_1(\boldsymbol{s}),\ldots,h_m(\boldsymbol{s}))$.

Definition 2 (Pareto-dominance Relation [26])**.** For two solutions $\boldsymbol{s}$ and $\boldsymbol{s}'$ in a multi-objective minimization problem (where $\boldsymbol{s},\boldsymbol{s}'\in\mathcal{X}$) with m objective functions $h_1,\ldots,h_m$,

(1) $\boldsymbol{s}$ weakly dominates $\boldsymbol{s}'$ (denoted as $\boldsymbol{s}\preceq\boldsymbol{s}'$) if and only if $\forall 1\le i\le m, h_i(\boldsymbol{s})\le h_i(\boldsymbol{s}')$;

(2) $\boldsymbol{s}$ dominates $\boldsymbol{s}'$ (denoted as $\boldsymbol{s}\prec\boldsymbol{s}'$) if and only if $\boldsymbol{s}\preceq\boldsymbol{s}'$ and $\exists 1\le i\le m, h_i(\boldsymbol{s})<h_i(\boldsymbol{s}')$.

If neither $\boldsymbol{s}$ nor $\boldsymbol{s}'$ can weakly dominate each other, we say that these two solutions are incomparable. Actually, the definition of Pareto-dominance relation in a multi-objective maximization problem is similar and we just need to swap the direction of inequalities in (1) and (2).

Definition 3 (Pareto Front [26])**.** A feasible solution of a multi-objective optimization problem which cannot be dominated by any other feasible solutions of this problem is called a Pareto optimal solution, where a feasible solution refers to a solution satisfying all the constraints of this problem. The set of the objective vectors of all the Pareto optimal solutions is called the Pareto front.

B. Subset Selection Problem in EMO

MOEAs are popular in solving multi-objective optimization problems and the subset selection problem is crucial for these algorithms, which brings the problem of subset selection in EMO. First, we formally define this problem as follows.

Definition 4 (Subset Selection Problem in EMO)**.** Given the set G of elements $\boldsymbol{s}\in\mathbb{R}^m$, the set quality indicator function

$f : 2^G \to \mathbb{R}$ used in EMO and a positive integer k, the subset selection problem is to find $\arg\max_{S\subseteq G}\ f(S)\ s.t.\ |S| \leq k$.

In brief, this problem is to select a subset with a cardinality constraint from a universe to maximize the quality indicator which is our evaluation function to the selected subset. Moreover, since some indicators are to be maximized (e.g., HV) and other indicators are to be minimized (e.g., IGD/IGD+/R2), we take the minus IGD/IGD+/R2 indicator as the objective function to evaluate the subset to be consistent with the problem definition. Note that we are only interested in the objective space, and thus the concept of solution and set mentioned in Definition 4 and the following paper refer to the point vector and set of points in the objective space, respectively.

C. Quality Indicators for Subset Selection Problem

Aimed at solving subset selection problem in the evolutionary multi-objective optimization, we choose four representative quality indicators for the subset quality evaluation from a number of state-of-the-art quality indicators [10]. It is worth mentioning [13, 27–30] that hypervolume is Pareto compliant (i.e., a better solution set in terms of Pareto dominance implies a better hypervolume value), IGD is not Pareto compliant, and IGD+ and R2 are weakly Pareto compliant (i.e., a better solution set implies a better or equivalent indicator value). Generally, this property means that to what degree can the indicator demonstrate the quality of solutions based on Pareto-dominance relation. (Weak) Pareto compliance is an important property that implies there is no situation that a solution set evaluated better would never be preferred by the decision maker under any circumstance. This unwelcome situation may happen, particularly in real-world scenarios where the decision maker's preferences are available [31].

1) Hypervolume: The hypervolume indicator measures the volume enclosed by the solution set and a reference point. It is the most famous quality indicator used in many applications because of its theoretical desirability such as being Pareto compatible and having intuitive geometric meaning. Hypervolume is defined formally as follows:

Definition 5 (Hypervolume Indicator [11])**.** Given a solution set S of elements $\boldsymbol{s} \in \mathbb{R}^m$ and a reference point $\boldsymbol{r} \in \mathbb{R}^m$, the value of hypervolume can be calculated as

$$HV(S) = \lambda\left(\cup_{\boldsymbol{s}\in S}\left\{\boldsymbol{x} | \boldsymbol{s} \prec \boldsymbol{x} \prec \boldsymbol{r}\right\}\right),$$

where λ denotes the Lebesgue measure.

2) Inverted Generational Distance (IGD): Similar to the hypervolume indicator, IGD is also one of the most widely used indicators. It measures the distance from the solution set to the Pareto front; a low value of this indicator is preferable. Intuitively, we can get a set close to the Pareto front which keeps diverse solutions by minimizing the value of IGD. Note that we always need a large reference set when using IGD. The formal definition of IGD is:

Definition 6 (IGD Indicator [12])**.** Given a solution set S of elements $\boldsymbol{s} \in \mathbb{R}^m$ and a reference set R of elements $\boldsymbol{r} \in \mathbb{R}^m$, the value of IGD is calculated by

$$IGD(S, R) = \frac{\sum_{\boldsymbol{r}\in R} \min_{\boldsymbol{s}\in S} dist_e(\boldsymbol{s}, \boldsymbol{r})}{|R|},$$

where $dist_e(\boldsymbol{s}, \boldsymbol{r})$ denotes the Euclidean distance between the solution $\boldsymbol{s}$ and the reference point $\boldsymbol{r}$. The formula of this distance function is as follows:

$$dist_e(\boldsymbol{s}, \boldsymbol{r}) = \sqrt{\sum_{i=1}^{m}(s_i - r_i)^2},$$

where m is the number of objectives.

3) IGD+: As the name suggested, IGD+ is a variant of IGD and the only difference between these two indicators is that IGD+ uses a different distance called IGD+ distance. For that reason, the following definition of this indicator is almost the same as IGD except the distance function:

Definition 7 (IGD+ Indicator [13])**.** Given a solution set S of elements $\boldsymbol{s} \in \mathbb{R}^m$ and a reference set R of elements $\boldsymbol{r} \in \mathbb{R}^m$, the value of IGD+ is calculated by

$$IGD_+(S, R) = \frac{\sum_{\boldsymbol{r}\in R} \min_{\boldsymbol{s}\in S} dist_p(\boldsymbol{s}, \boldsymbol{r})}{|R|},$$

and the formula of IGD+ distance function $dist_p(\boldsymbol{s}, \boldsymbol{r})$ is defined as

$$dist_p(\boldsymbol{s}, \boldsymbol{r}) = \sqrt{\sum_{i=1}^{m}\left(\max\left\{0, s_i - r_i\right\}\right)^2},$$

where m is the number of objectives.

4) R2: Different from all indicators above, the R2 indicator can take the preference into consideration by using different weight vectors. Besides, the R2 indicator uses the utility function to significantly accelerate the calculation. These two useful properties make the R2 indicator a commonly used quality indicator in multi-objective optimization. Note that most previous studies use the standard weighted Tchebycheff function [30, 32–34].

Definition 8 (R2 Indicator with Weighted Tchebycheff Function [14])**.** Given a solution set S of elements $\boldsymbol{s} \in \mathbb{R}^m$, a set W of weight vectors $\boldsymbol{w} \in \mathbb{R}^m$ and a Utopian point $\boldsymbol{r} \in \mathbb{R}^m$, the R2 indicator with the standard weighted Tchebycheff function is defined as follows:

$$R_2(S, W, \boldsymbol{r}) = \frac{\sum_{\boldsymbol{w}\in W} \min_{\boldsymbol{s}\in S} Tch(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{r})}{|W|},$$

and the standard weighted Tchebycheff function is defined as

$$Tch(\boldsymbol{s}, \boldsymbol{w}, \boldsymbol{r}) = \max_{i\in\{1,\ldots,m\}}\{w_i|r_i - s_i|\},$$

where $\boldsymbol{w} = (w_1, w_2, ..., w_m)$ is a given weight vector with $\|\boldsymbol{w}\|_1 = 1$ and $w_1, w_2, ..., w_m \geq 0$.

D. Previous Algorithms

In the field of subset selection, greedy algorithms are popular among all practical algorithms. For this problem, Chen et al. [16] proposed a greedy algorithm which "lazily" updates the marginal gain of the hypervolume value. After that, Chen et al. [19] applied the conclusion in [16] to IGD and IGD+ indicators and further provided an acceleration method for the calculation of these indicators.

According to the seminal work in [18], the classic greedy algorithm achieves the state-of-the-art $(1-1/e)$-approximation when the quality indicator function is monotone and submodular. This conclusion shows that the greedy algorithms in [16, 19] can get a solution of which the quality indicator value is at least a $(1-1/e)$ multiple of the quality indicator value of an optimal solution. As a matter of fact, these two properties of set functions are common among many quality indicators and we introduce them as follows.

Definition 9 (Monotone)**.** A set function $f : 2^G \to \mathbb{R}$ is monotone if and only if $f(B) \geq f(A),\ \forall A \subseteq B \subseteq G$.

Definition 10 (Submodular)**.** A set function $f : 2^G \to \mathbb{R}$ is submodular if and only if $f(e|A) \geq f(e|B),\ \forall A \subseteq B \subseteq G$ and $e \notin B$, where $f(e|A) = f(A \cup \{e\}) - f(A)$.

Intuitively, submodularity refers to the concept of diminishing returns, i.e., the benefit of adding an element to a set will not increase as the set extends.

III. Algorithmic Framework and Algorithms

In this section, we elaborate on our algorithmic framework and simple MOEAs which accelerate the calculation of quality indicators. Motivated by [22], we propose a multi-objective evolutionary algorithmic framework for the subset selection problem in EMO. Firstly, we formulate this problem into a bi-objective problem and use a MOEA to solve it. Unlike the problem in Definition 4, we set the subset size as one extra objective instead of a cardinality constraint. The formal definition of the bi-objective problem is as follows:

Definition 11 (Bi-objective Form of Subset Selection Problem in EMO)**.** Given the set G of elements $\boldsymbol{s} \in \mathbb{R}^m$, the quality indicator function $f : 2^G \to \mathbb{R}$ used in EMO and a positive integer k, the problem is to find optimal solutions of the following multi-objective optimization problem

$$\arg\max_{S \subseteq G}\ g_i(S),\ \forall i \in \{1, 2\},$$

where the two objectives g_1, g_2 are set as

$$g_1(S) = \begin{cases} -\infty, & S = \{0\}^n,\ \text{or}\ |S| \geq 2k \\ f(S), & \text{otherwise} \end{cases}, \quad g_2(S) = -|S|.$$

Here we use $2k$ to bound the set size in order to improve the efficiency of our algorithms, which can be flexibly adjusted in practice. Notably, any MOEA can be utilized to solve this bi-objective problem.

With this framework, we design several algorithms based on GSEMO for each of the representative quality indicators. For the IGD/IGD+/R2 indicators, our algorithms accelerate the calculation procedure of these indicators by using their properties. In the following subsections, we will explain our algorithms in more details. We first introduce the framework of GSEMO for solving the subset selection problem in EMO and then describe our algorithms accordingly.

A. GSEMO Algorithm

The performance of the greedy algorithm sometimes is affected by its greedy nature. In order to avoid falling into local optima at all possible, we propose our algorithms based on GSEMO, a simple evolutionary algorithm firstly invented by Laumanns et al. [23] and widely used in theoretical analyses [23, 35–37]. We give a brief introduction about the framework of GSEMO in our problem and the procedure of this algorithm is shown in Algorithm 1.

Algorithm 1 GSEMO

Input: the ground set G, the quality indicator function f and an integer parameter $k \in [1, n]$
Parameter: the number T of iterations
Output: a subset of G with at most k elements
Process:

```
1: Let x = {0}^n, P = {x} and t = 0.
2: while t < T do
3:    Select x from P uniformly at random.
4:    Generate x' by flipping each bit of x with prob. 1/n.
5:    if ∄y ∈ P such that y ≻ x' then
6:       P = (P\{y ∈ P | x' ⪰ y}) ∪ {x'}
7:    end if
8:    t = t + 1
9: end while
10: return arg max_{x∈P,|x|≤k} f(x)
```

To start with, we give some explanations about notations in Algorithm 1. For the simplicity of representation, we use the binary vector $\boldsymbol{x} \in \{0, 1\}^n$ to indicate whether each element of the ground set G is in a subset X, i.e., the i-th element belongs to X if and only if the i-th bit $x_i = 1$. In this paper, we will not distinguish between these two symbols, i.e., $\boldsymbol{x}$ and X. Besides, as shown in Definition 11, the two objectives g_1, g_2 in this problem are the quality indicator and the minus of the subset size, respectively. Moreover, in line 5 of Algorithm 1, $\boldsymbol{y} \succ \boldsymbol{x}'$ equals to $(g_1(\boldsymbol{y}) > g_1(\boldsymbol{x}')) \wedge (g_2(\boldsymbol{y}) \geq g_2(\boldsymbol{x}'))$ or $(g_1(\boldsymbol{y}) \geq g_1(\boldsymbol{x}')) \wedge (g_2(\boldsymbol{y}) > g_2(\boldsymbol{x}'))$ according to the Pareto-dominance relation in Definition 2. Similarly, $\boldsymbol{x}' \succeq \boldsymbol{y}$ in line 6 of Algorithm 1 equals to $(g_1(\boldsymbol{x}') \geq g_1(\boldsymbol{y})) \wedge (g_2(\boldsymbol{x}') \geq g_2(\boldsymbol{y}))$.

GSEMO begins with the empty set and tries to find better solutions and caches them in the population P in T iterations. In each iteration, it will generate a new set $\boldsymbol{x}'$ by flipping each bit of a set $\boldsymbol{x} \in P$ with probability $1/n$ and compare $\boldsymbol{x}'$ with the sets in P. If this newly generated set $\boldsymbol{x}'$ is not dominated by any set in P, it will be added into P and those sets weakly dominated by it will be deleted from P. Obviously the solution sets in the population P cannot dominate each other. Note that for those non-dominated sets with the same objective vector, only the latest one is kept in the population P.

B. Proposed Algorithms

Now we introduce our algorithms for the subset selection problem in EMO with respect to representative quality indicators (i.e., hypervolume, IGD, IGD+, R2 indicator). For the hypervolume indicator, we simply apply GSEMO. For the rest quality indicators, we design an accelerated variant of GSEMO (i.e., GSEMO-ACC) as presented in Algorithm 2 and customize subroutines of GSEMO-ACC for each of these indicators to accelerate the calculation of the indicator value.

Algorithm 2 differs from GSEMO shown in Algorithm 1 mainly because of the additional part of the subroutine (i.e., ACC in line 9 of Algorithm 2) which uses an acceleration method for calculating the quality indicator values. The idea of our method to accelerate is inspired by Chen et al. [19]. Specifically, Chen et al. proposed a method to accelerate the calculation of the IGD improvement. They use an array to cache temporal results and reduce the time complexity of this calculation from $O(m|X||R|)$ to $O(m|R|)$, where X and R are the solution set and reference set, respectively.

Our acceleration method maintains a vector $\boldsymbol{v_x}$ for each solution $\boldsymbol{x}$ in the current population P, which involves the component values of the quality indicator with respect to the corresponding solution $\boldsymbol{x}$. Concretely, in the calculation of IGD/IGD+ indicator, the vector $\boldsymbol{v_x}$ corresponding to a solution $\boldsymbol{x}$ contains $|R|$ elements which are the minimum distances between each reference point $\boldsymbol{r} \in R$ and elements in $\boldsymbol{x}$. Similarly, in the calculation of R2 indicator, $\boldsymbol{v_x}$ contains $|W|$ elements which are the minimum Tchebycheff values between each weight vector $\boldsymbol{w} \in W$ and elements in $\boldsymbol{x}$. In these two cases, $\boldsymbol{v_x}$ is called a distance or Tchebycheff value vector. When $\boldsymbol{x}$ is an empty set, i.e., $\{0\}^n$, it is obvious that the elements of $\boldsymbol{v_x}$ are the infinity. By the vector $\boldsymbol{v_x}$, the quality indicator value of $\boldsymbol{x}$ is just the mean of the elements of $\boldsymbol{v_x}$, as shown in line 11 of Algorithm 2.

1) Hypervolume Subset Selection Problem: As the name suggested, the hypervolume subset selection problem uses the hypervolume indicator as the selection criterion of the subset selection problem with a cardinality constraint. We aim to select a subset with the maximal hypervolume value from candidate subsets. For this problem, we use GSEMO displayed in Algorithm 1. Since we use the hypervolume indicator as the objective function, we name it GSEMO-HV.

2) IGD/IGD+ Subset Selection Problem: This problem is similar to the hypervolume subset selection problem, except that it uses IGD/IGD+ to indicate the selection procedure and a low value of these indicators is preferable. As mentioned above, we take the minus IGD/IGD+ indicator as the objective function to evaluate the subset.

For this problem, we use GSEMO-ACC as described in Algorithm 2. We set the quality indicator f as the minus IGD/IGD+ indicator and the set R as the reference set. Besides, the vector $\boldsymbol{v_x}$ stores the distances between each reference point and the corresponding nearest element in $\boldsymbol{x}$. To be accordance with GSEMO-HV, we name our algorithms as GSEMO-ACC-IGD/GSEMO-ACC-IGD+ according to the quality indicator. Then we design the specific subroutines (i.e., ACC-IGD/ACC-IGD+) for these two algorithms respectively, which act as the role of the subroutine ACC in Algorithm 2.

Algorithm 2 GSEMO-ACC

Input: the ground set G, the quality indicator function f and an integer parameter $k \in [1, n]$
Parameter: the number T of iterations and the set R used for calculating f
Output: a subset of G with at most k elements
Process:
1: Let $\boldsymbol{x} = \{0\}^n$, $P = \{\boldsymbol{x}\}$ and $t = 0$.
2: Let $\boldsymbol{v_x} = \{+\infty\}^{|R|}$.
3: **while** $t < T$ **do**
4: Select $\boldsymbol{x}$ from P uniformly at random.
5: Generate $\boldsymbol{x'}$ by flipping each bit of $\boldsymbol{x}$ with prob. $1/n$.
6: **if** $\boldsymbol{x'} = \{0\}^n$ **then**
7: $\boldsymbol{v_{x'}} = \{+\infty\}^{|R|}$
8: **else**
9: $\boldsymbol{v_{x'}} = \text{ACC}(\boldsymbol{v_x}, \boldsymbol{x}, \boldsymbol{x'}, R)$
10: **end if**
11: $g_1(\boldsymbol{x'}) = \text{mean}(\boldsymbol{v_{x'}})$.
12: **if** $\nexists \boldsymbol{y} \in P$ such that $\boldsymbol{y} \succ \boldsymbol{x'}$ **then**
13: $P = (P \backslash \{\boldsymbol{y} \in P | \boldsymbol{x'} \succeq \boldsymbol{y}\}) \cup \{\boldsymbol{x'}\}$
14: **end if**
15: $t = t + 1$
16: **end while**
17: **return** $\arg\max_{\boldsymbol{x} \in P, |\boldsymbol{x}| \leq k} f(\boldsymbol{x})$

As mentioned above, Chen et al. [19] proposed a fast variant of greedy algorithm to accelerate the calculation of the improvement of the IGD/IGD+ indicator by using an array to cache temporary results, and they reduced the time of this calculation from $O(m|X||R|)$ to $O(m|R|)$, where m is the number of objectives and X, R represent the candidate solution set X and the reference set R, respectively. Our acceleration method in ACC-IGD/ACC-IGD+ is motivated by this idea, which also reduces the time of IGD/IGD+ calculation to $O(m|R|)$. Besides, we accelerate the calculation of the R2 indicator from $O(m|X||W|)$ to $O(m|W|)$. However, our algorithm and the corresponding theoretical analysis are more complicated, because we have to consider the operations of adding and deleting an element from the candidate solution set while the previous work [19] only needs to handle with the former operation due to its greedy characteristic.

Since ACC-IGD and ACC-IGD+ have exactly the same algorithmic framework and the only difference between them is that they use different distance functions, we only present ACC-IGD for the sake of simplicity. According to Definition 6, the calculation of IGD value with respect to a newly generated solution $\boldsymbol{x'}$ requires $O(m|\boldsymbol{x'}||R|)$ time. To accelerate, we leverage the property that the IGD value of $\boldsymbol{x'}$ with regard to the reference set R is the mean value of the corresponding distance vector $\boldsymbol{v_{x'}}$. Instead of calculating $\boldsymbol{v_{x'}}$ from scratch, we derive it from $\boldsymbol{v_x}$ by considering the adding and deleting operations, respectively. ACC-IGD is described in Algorithm 3, in which a subroutine (i.e., UPDATE-DIST) is employed to handle the updating procedure of the distance vector as shown in Algorithm 4.

Since we want to accelerate the calculation of IGD value of

Algorithm 3 ACC-IGD

Input: the distance vector $\boldsymbol{v_x}$, the sets $\boldsymbol{x}$, $\boldsymbol{x'}$ and the reference set R
Output: the distance vector $\boldsymbol{v_{x'}}$
Process:
1: $S_{\rm add} = \boldsymbol{x'} \backslash \boldsymbol{x}$.
2: $\boldsymbol{v_{x'}} =$ UPDATE-DIST$(\boldsymbol{v_x}, S_{\rm add}, R)$.
3: $S_{\rm del} = \boldsymbol{x} \backslash \boldsymbol{x'}$.
4: $R_{\rm re} = \emptyset$.
5: **for** $\boldsymbol{e}$ in $S_{\rm del}$ **do**
6: $\quad \boldsymbol{v_e} =$ Euclidean distance from $\boldsymbol{e}$ to each point in R.
7: $\quad R_{\rm re} = R_{\rm re} \cup \{\boldsymbol{r} | \boldsymbol{r} \in R, \boldsymbol{v_e}[\boldsymbol{r}] = \boldsymbol{v_{x'}}[\boldsymbol{r}]\}$
8: **end for**
9: **for** $\boldsymbol{r}$ in $R_{\rm re}$ **do**
10: $\quad \boldsymbol{v_{x'}}[\boldsymbol{r}] =$ UPDATE-DIST$(+\infty, \boldsymbol{x'}, \boldsymbol{r})$
11: **end for**
12: **return** $\boldsymbol{v_{x'}}$

Algorithm 4 UPDATE-DIST: Subroutine of ACC-IGD

Input: the distance vector $\boldsymbol{v}$, the set S and the reference set R
Output: a distance vector
Process:
1: **for** $\boldsymbol{e}$ in S **do**
2: $\quad \boldsymbol{v_e} =$ Euclidean distance from $\boldsymbol{e}$ to each point in R.
3: $\quad \boldsymbol{v'} = \min\{\boldsymbol{v}, \boldsymbol{v_e}\}$
4: **end for**
5: **return** $\boldsymbol{v'}$

the newly generated solution $\boldsymbol{x'}$ by ACC-IGD and this solution is generated by the bit-wise mutation procedure in line 5 of GSEMO-ACC, we study this vital procedure. As the mutation procedure may randomly add some elements to the selected candidate solution $\boldsymbol{x}$ or delete some elements from it, we design two methods to deal with these two cases, respectively. In ACC-IGD, we use $S_{\rm add}$ (line 1 of Algorithm 3) to represent the set of elements newly added to $\boldsymbol{x}$ and included in $\boldsymbol{x'}$. Similarly, $S_{\rm del}$ (line 3 of Algorithm 3) represents the set of elements deleted from $\boldsymbol{x}$ and absent in $\boldsymbol{x'}$.

When an element $\boldsymbol{e}$ is added to $\boldsymbol{x}$, i.e., $\boldsymbol{e} \in S_{\rm add}$, we only need to calculate the distance between $\boldsymbol{e}$ and each reference point and store them in a new vector $\boldsymbol{v_e}$ (line 2 of Algorithm 4). Then we update the distance vector $\boldsymbol{v_{x'}}$ by taking the minimum value between $\boldsymbol{v_{x'}}$ and $\boldsymbol{v_e}$ through line 3 of Algorithm 4, where the operator $\min$ is the point-wise minimization. To be specific, by this operator we generate a new vector $\boldsymbol{v'}$ which has the same number of dimensions with $\boldsymbol{v}$ and $\boldsymbol{v_e}$, and the value of $\boldsymbol{v'}$ on each dimension is the minimum value of $\boldsymbol{v}$ and $\boldsymbol{v_e}$ on the corresponding dimension.

However, when some element $\boldsymbol{e}$ is deleted from $\boldsymbol{x}$, i.e., $\boldsymbol{e} \in S_{\rm del}$, we give a more delicate method with the basic idea of reducing redundant calculations by carrying out deleting operation cautiously. Firstly, we calculate $\boldsymbol{v_e}$ just as the former case and compare $\boldsymbol{v_e}$ with the previous distance vector $\boldsymbol{v_{x'}}$ in lines 6 and 7 of Algorithm 3. We put all the reference points $\boldsymbol{r}$ which satisfy that $\boldsymbol{v_e}[\boldsymbol{r}]$ is equal to $\boldsymbol{v_{x'}}[\boldsymbol{r}]$ into the set $R_{\rm re}$. Note that $\boldsymbol{v_e}[\boldsymbol{r}]$ represents the element of the vector $\boldsymbol{v_e}$ corresponding to the reference point $\boldsymbol{r}$, and $\boldsymbol{v_{x'}}[\boldsymbol{r}]$ is alike. For each reference point in $R_{\rm re}$, we re-calculate the minimum distance from it to the new set $\boldsymbol{x'}$ in line 10 of Algorithm 3, which can also be accomplished by using the subroutine UPDATE-DIST in Algorithm 4.

By the proposed acceleration method ACC-IGD, the time complexity becomes $O(m|S_{\rm add}||R|) + O(m|S_{\rm del}||R|) + O(m|R_{\rm re}||\boldsymbol{x'}|)$, the three terms of which correspond to lines 2, 5–8 and 9–11 of Algorithm 3, respectively. As introduced above, $S_{\rm add}$ and $S_{\rm del}$ contain the added and deleted elements when generating $\boldsymbol{x'}$ from $\boldsymbol{x}$, and $R_{\rm re}$ contains the reference points whose distances need to re-calculate. Though this complexity seems complicated, we will theoretically show that it is actually $O(m|R|)$ in expectation in the next subsection, which is significantly lower than the complexity $O(m|R||\boldsymbol{x'}|)$ of directly calculating $\boldsymbol{v_{x'}}$.

3) R2 Subset Selection Problem: The R2 subset selection problem resembles the IGD/IGD+ subset selection problem, since we can find that R2 indicator has a similar form to IGD/IGD+ indicators and the calculation of each of these indicators is to get the mean value of some minima. So, we are able to use GSEMO-ACC in this problem and we just need to make a slight change on the parameters. Specifically, we replace the reference set R with the set W of weight vectors and consider another parameter, i.e., the Utopian point $\boldsymbol{r}$. Because a low value of R2 indicator is also preferable, we use the minus R2 indicator as the objective function for subset evaluation just the same as before. In brief, we use a vector $\boldsymbol{v_x}$ to store the minimum Tchebycheff values with respect to each weight vector $\boldsymbol{w}$ and the corresponding element in $\boldsymbol{x}$.

Algorithm 5 ACC-R2

Input: the Tchebycheff value vector $\boldsymbol{v_x}$, the sets $\boldsymbol{x}$, $\boldsymbol{x'}$, the set W of weight vectors and the Utopian point $\boldsymbol{r}$
Output: the Tchebycheff value vector $\boldsymbol{v_{x'}}$
Process:
1: $S_{\rm add} = \boldsymbol{x'} \backslash \boldsymbol{x}$.
2: $\boldsymbol{v_{x'}} =$ UPDATE-TCH$(\boldsymbol{v_x}, S_{\rm add}, W, \boldsymbol{r})$.
3: $S_{\rm del} = \boldsymbol{x} \backslash \boldsymbol{x'}$.
4: $W_{\rm re} = \emptyset$.
5: **for** $\boldsymbol{e}$ in $S_{\rm del}$ **do**
6: $\quad \boldsymbol{v_e} = \{Tch(\boldsymbol{e}, \boldsymbol{w}, \boldsymbol{r}) | \boldsymbol{w} \in W\}$.
7: $\quad W_{\rm re} = W_{\rm re} \cup \{\boldsymbol{w} | \boldsymbol{w} \in W, \boldsymbol{v_e}[\boldsymbol{w}] = \boldsymbol{v_{x'}}[\boldsymbol{w}]\}$
8: **end for**
9: **for** $\boldsymbol{w}$ in $W_{\rm re}$ **do**
10: $\quad \boldsymbol{v_{x'}}[\boldsymbol{w}] =$ UPDATE-TCH$(+\infty, \boldsymbol{x'}, \boldsymbol{w}, \boldsymbol{r})$
11: **end for**
12: **return** $\boldsymbol{v_{x'}}$

We name this algorithm GSEMO-ACC-R2 and offer a customized subroutine (i.e., ACC-R2) for it. ACC-R2 is described in Algorithm 5 and a subroutine (i.e., UPDATE-TCH) is used to handle the updating procedure of the Tchebycheff value vector as shown in Algorithm 6. The framework of ACC-R2 is similar to ACC-IGD.

Algorithm 6 UPDATE-TCH: Subroutine of ACC-R2

Input: the Tchebcheff value vector $\boldsymbol{v}$, the set S, the set W of weight vectors and the Utopian point $\boldsymbol{r}$
Output: a Tchebycheff value vector
Process:
1: **for** $\boldsymbol{e}$ in S **do**
2: $\quad \boldsymbol{v_e} = \{Tch(\boldsymbol{e}, \boldsymbol{w}, \boldsymbol{r}) | \boldsymbol{w} \in W\}$.
3: $\quad \boldsymbol{v}' = \min\{\boldsymbol{v}, \boldsymbol{v_e}\}$
4: **end for**
5: **return** $\boldsymbol{v}'$

C. Theoretical Analysis of Acceleration

Next, we give the theoretical analysis about the time complexity of our customized subroutines (i.e., ACC-IGD/IGD+ and ACC-R2). First of all, we give a mathematical conclusion in Lemma 1 to help with our theoretical analysis.

Lemma 1. *For $x \geq 1$ and non-negative integer L, we have*

$$\sum_{t=0}^{L} \binom{L}{t} \left(\frac{1}{x}\right)^t \left(1 - \frac{1}{x}\right)^{L-t} \cdot O(tx) = O(L).$$

Proof. We represent the left side of the statement as M and we have

$$M = O\left(\sum_{t=1}^{L} t\binom{L}{t} \left(\frac{1}{x}\right)^{t-1} \left(1 - \frac{1}{x}\right)^{L-t}\right).$$

Because

$$t\binom{L}{t} = L\frac{(L-1)!}{(t-1)!\,((L-1)-(t-1))!} = L\binom{L-1}{t-1},$$

we have

$$\begin{aligned} M &= O\left(L\sum_{t=1}^{L} \binom{L-1}{t-1} \left(\frac{1}{x}\right)^{t-1} \left(1 - \frac{1}{x}\right)^{(L-1)-(t-1)}\right) \\ &= O\left(L\sum_{t=0}^{L-1} \binom{L-1}{t} \left(\frac{1}{x}\right)^{t} \left(1 - \frac{1}{x}\right)^{(L-1)-t}\right). \end{aligned}$$

By the binomial theorem, we can conclude that $M = O(L)$. ■

By analyzing the three main components (i.e., lines 2, 5–8, and 9–11 in Algorithm 3) of ACC-IGD, we prove in Theorem 1 that the expected time complexity of ACC-IGD is $O(m|R|)$. Line 2 corresponds to the process of updating the distance value vector when adding elements, while lines 5–8 and 9–11 correspond to the process when deleting elements. Similarly, the expected time complexity of ACC-IGD+ and ACC-R2 can be proved to be $O(m|R|)$ and $O(m|W|)$, respectively. In the following analysis, we use $\mathbb{E}[\cdot]$ to denote the expectation of a random variable.

Theorem 1. *The expected time complexity of ACC-IGD/IGD+ and ACC-R2 is $O(m|R|)$ and $O(m|W|)$, respectively, where m is the number of objectives, R is the reference set used for calculating IGD/IGD+, and W is the set of weight vectors used for calculating R2.*

Proof. We first analyze the time complexity of ACC-IGD (i.e., Algorithm 3), which mainly depends on the time complexity of lines 2, 5–8, and 9–11. Line 2 of Algorithm 3 corresponds to updating the distance vector when adding elements. The set of added elements is denoted as $S_{\rm add}$. When an element $\boldsymbol{e} \in S_{\rm add}$ is added to $\boldsymbol{x}$, we only need to calculate the distance between $\boldsymbol{e}$ and each reference point in R (i.e., $\boldsymbol{v_e}$ in line 2 of Algorithm 4), whose time complexity is $O(m|R|)$ as each vector is m-dimensional. Thus, $O(m|S_{\rm add}||R|)$ time is required by line 2 of Algorithm 3.

Lines 5–8 and 9–11 of Algorithm 3 correspond to updating the distance vector when deleting elements. The set of deleted elements is denoted as $S_{\rm del}$. When an element $\boldsymbol{e} \in S_{\rm del}$ is deleted from $\boldsymbol{x}$, line 6 also calculates the vector $\boldsymbol{v_e}$ containing the distance between $\boldsymbol{e}$ and each reference point in R; thus, the time complexity of lines 5–8 is $O(m|S_{\rm del}||R|)$. The deletion of an element $\boldsymbol{e}$ will lead to re-calculating the minimum distance of a reference point $\boldsymbol{r}$ w.r.t. the elements in $\boldsymbol{x} \cup \boldsymbol{x}'$, when $\boldsymbol{v_e}[\boldsymbol{r}] = \boldsymbol{v_{x'}}[\boldsymbol{r}]$ as in line 7 of Algorithm 3. That is, when the nearest element to the reference point $\boldsymbol{r}$ is deleted, a re-calculation of the minimum distance is needed. The set of reference points requiring re-calculation is denoted as $R_{\rm re}$. Because the calculation of the minimum distance between a reference point $\boldsymbol{r}$ and the elements in $\boldsymbol{x}'$ (i.e., line 10) requires $O(m|\boldsymbol{x}'|)$ time, the time complexity of lines 9–11 of Algorithm 3 is $O(m|R_{\rm re}||\boldsymbol{x}'|)$.

Based on the above analysis, the total time complexity of ACC-IGD is

$$O(m|S_{\rm add}||R|) + O(m|S_{\rm del}||R|) + O(m|R_{\rm re}||\boldsymbol{x}'|), \quad (1)$$

given $S_{\rm add}$, $S_{\rm del}$ and $R_{\rm re}$. However, $S_{\rm add}$, $S_{\rm del}$ and $R_{\rm re}$ are random variables because $\boldsymbol{x}'$ is generated from $\boldsymbol{x}$ by bit-wise mutation in line 5 of Algorithm 2. The number $|S_{\rm add}|$ of added elements is just the number of 0-bits of $\boldsymbol{x}$ flipped to 1-bits by bit-wise mutation. Similarly, the number $|S_{\rm del}|$ of deleted elements is the number of 1-bits of $\boldsymbol{x}$ flipped to 0-bits. Let $|\boldsymbol{x}|$ denote the number of 1-bits of $\boldsymbol{x}$. As each bit is flipped independently with probability $1/n$ by bit-wise mutation, where n denotes the size of the ground set G, i.e., the length of the Boolean vector representation, we have

$$\mathbb{E}[|S_{\rm add}|] = 1 - |\boldsymbol{x}|/n; \qquad \mathbb{E}[|S_{\rm del}|] = |\boldsymbol{x}|/n. \quad (2)$$

Given that the number $|S_{\rm del}|$ of deleted elements is i, where $0 \leq i \leq |\boldsymbol{x}|$, we now analyze the probability that the minimum distance of a reference point $\boldsymbol{r}$ needs to be re-calculated. Note that the current minimum distance of the reference point $\boldsymbol{r}$ is based on the nearest element in $\boldsymbol{x}' \cup \boldsymbol{x}$. If the nearest element is in $\boldsymbol{x}' \setminus \boldsymbol{x}$, i.e., $S_{\rm add}$, the probability is obviously 0, because the deleted elements are from $\boldsymbol{x}$. Otherwise, the probability is $i/|\boldsymbol{x}|$, because now i elements are randomly deleted from $\boldsymbol{x}$. Note that for the simplicity, we have made an assumption that for a given reference point, its distance from each element is different. As the distance calculation involves operations between multi-dimensional vectors, the assumption can hold in most cases. According to this probability, we can calculate the distribution of $|R_{\rm re}|$, i.e., the number of the reference points

whose minimum distances need to be re-calculated. That is,

$$\forall j \geq 1 : P(|R_{\mathrm{re}}| = j) \leq \sum_{i=0}^{|\boldsymbol{x}|} P(|S_{\mathrm{del}}| = i) \cdot \binom{|R|}{j} \left(\frac{i}{|\boldsymbol{x}|}\right)^{j} \left(1 - \frac{i}{|\boldsymbol{x}|}\right)^{|R|-j}.$$

Then, we can derive an upper bound on the expectation of $|R_{\mathrm{re}}|$ as follows.

$$\begin{aligned}
&\mathbb{E}[|R_{\mathrm{re}}|] = \sum_{j=0}^{|R|} j \cdot P(|R_{\mathrm{re}}| = j) \\
&\leq \sum_{j=0}^{|R|} j \cdot \sum_{i=0}^{|\boldsymbol{x}|} P(|S_{\mathrm{del}}| = i) \binom{|R|}{j} \left(\frac{i}{|\boldsymbol{x}|}\right)^{j} \left(1 - \frac{i}{|\boldsymbol{x}|}\right)^{|R|-j} \\
&= \sum_{i=0}^{|\boldsymbol{x}|} P(|S_{\mathrm{del}}| = i) \cdot \sum_{j=0}^{|R|} j \binom{|R|}{j} \left(\frac{i}{|\boldsymbol{x}|}\right)^{j} \left(1 - \frac{i}{|\boldsymbol{x}|}\right)^{|R|-j} \\
&= \sum_{i=0}^{|\boldsymbol{x}|} P(|S_{\mathrm{del}}| = i) \cdot \frac{i}{|\boldsymbol{x}|} \cdot \\
&\quad \sum_{j=0}^{|R|} \binom{|R|}{j} \left(\frac{i}{|\boldsymbol{x}|}\right)^{j} \left(1 - \frac{i}{|\boldsymbol{x}|}\right)^{|R|-j} \cdot \frac{j|\boldsymbol{x}|}{i} \\
&= \sum_{i=0}^{|\boldsymbol{x}|} P(|S_{\mathrm{del}}| = i) \cdot (i/|\boldsymbol{x}|) \cdot O(|R|) \\
&= (\mathbb{E}[|S_{\mathrm{del}}|]/|\boldsymbol{x}|) \cdot O(|R|) = O(|R|/n),
\end{aligned}$$

where the penultimate equality holds by Lemma 1, and the last equality holds by Eq. (2). Combining Eq. (1), Eq. (2) and the above upper bound $O(|R|/n)$ on $\mathbb{E}[|R_{\mathrm{re}}|]$, we can conclude that the expected time complexity of ACC-IGD is

$$\begin{aligned}
&O(m|R|) \cdot (\mathbb{E}[|S_{\mathrm{add}}|] + \mathbb{E}[|S_{\mathrm{del}}|]) + O(m|\boldsymbol{x}'|) \cdot \mathbb{E}[|R_{\mathrm{re}}|] \\
&= O(m|R|) \cdot 1 + O(m|\boldsymbol{x}'|) \cdot O(|R|/n) = O(m|R|),
\end{aligned}$$

where the last equality holds by $|\boldsymbol{x}'| \leq n$.

The only difference between ACC-IGD and ACC-IGD+ is the employed distance between a reference point and an element (i.e., a solution of the multi-objective optimization problem) in the ground set G. ACC-IGD employs the Euclidean distance, while ACC-IGD+ uses the IGD+ distance. Obviously, this difference will not affect the time complexity analysis. That is, the time complexity of ACC-IGD+ is also $O(m|R|)$.

ACC-R2 in Algorithm 5 updates the Tchebycheff value vector instead of the distance value vector. As introduced in Section III-B, the distance value vector $\boldsymbol{v_x}$ for IGD contains $|R|$ elements which are the minimum distances between each reference point $\boldsymbol{r} \in R$ and elements in $\boldsymbol{x}$, while the Tchebycheff value vector for R2 contains $|W|$ elements which are the minimum Tchebycheff values between each weight vector $\boldsymbol{w} \in W$ and elements in $\boldsymbol{x}$. As the procedure of ACC-R2 in Algorithm 5 is similar to that of ACC-IGD in Algorithm 3, we can apply the same analysis procedure, except replacing the reference set R with the set W of weight vectors, to derive the expected time complexity $O(m|W|)$ of ACC-R2. ■

The time complexity of directly calculating the distance (or Tchebycheff) value vector $\boldsymbol{v}_{\boldsymbol{x}'}$ of the offspring solution $\boldsymbol{x}'$ is $O(m|\boldsymbol{x}'||R|)$ (or $O(m|\boldsymbol{x}'||W|)$), which requires calculating the distance between each reference point $\boldsymbol{r} \in R$ (or each weight vector $\boldsymbol{w} \in W$) and each element in $\boldsymbol{x}'$. Thus, combining Theorem 1, the following statement holds.

Remark 1. By the proposed accelerating method ACC-IGD/IGD+/R2, the time complexity of computing the IGD/IGD+/R2 indicator of the offspring solution $\boldsymbol{x}'$ can be reduced by a factor of $|\boldsymbol{x}'|$ in expectation.

IV. Theoretical Analysis

In this section, we prove the approximation guarantees of the proposed GSEMO-HV and GSEMO-ACC-IGD/IGD+/R2 for subset selection in EMO. The proof relies on the monotone and submodular property of the objective function, i.e., hypervolume or minus IGD/IGD+/R2. Because the hypervolume indicator and the minus IGD/IGD+ indicator have been proved to be monotone and submodular by Ulrich and Thiele [25] and Chen et al. [19], respectively, we only prove the monotonicity and submodularity of the minus R2 indicator here. Note that the concept of submodularity for hypervolume indicator subset selection was also discussed in [38], which summarized the performance of various approximation algorithms for the hypervolume subset selection problem such as the decremental greedy approach (gHSSD), the incremental greedy approach (gHSS), GSEMO and EPTAS (Efficient Polynomial-time Approximation Scheme) [24].

Theorem 2. *The minus R2 indicator is monotone (non-decreasing) and submodular.*

Proof. We begin with proving the monotonicity of the minus R2 indicator. Since $\min_{\boldsymbol{a} \in A \cup \{\boldsymbol{x}\}} Tch(\boldsymbol{a}, \boldsymbol{w}, \boldsymbol{r}) \leq \min_{\boldsymbol{a} \in A} Tch(\boldsymbol{a}, \boldsymbol{w}, \boldsymbol{r})$, we have

$$\begin{aligned}
R_2(A \cup \{\boldsymbol{x}\}, W, \boldsymbol{r}) &= \frac{1}{|W|} \sum_{\boldsymbol{w} \in W} \min_{\boldsymbol{a} \in A \cup \{\boldsymbol{x}\}} Tch(\boldsymbol{a}, \boldsymbol{w}, \boldsymbol{r}) \\
&\leq \frac{1}{|W|} \sum_{\boldsymbol{w} \in W} \min_{\boldsymbol{a} \in A} Tch(\boldsymbol{a}, \boldsymbol{w}, \boldsymbol{r}) \\
&= R_2(A, W, \boldsymbol{r}),
\end{aligned}$$

implying that the minus R2 indicator is monotone.

Next, we prove the submodularity of the minus R2 indicator. Based on the definition of submodularity, i.e., Definition 10, we need to prove that $\forall \boldsymbol{x}, \boldsymbol{y} \notin A$,

$$\begin{aligned}
&(-R_2(A \cup \{\boldsymbol{y}\}, W, \boldsymbol{r})) - (-R_2(A, W, \boldsymbol{r})) \\
&\geq (-R_2(A \cup \{\boldsymbol{x}\} \cup \{\boldsymbol{y}\}, W, \boldsymbol{r})) - (-R_2(A \cup \{\boldsymbol{x}\}, W, \boldsymbol{r})),
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
&R_2(A, W, \boldsymbol{r}) + R_2(A \cup \{\boldsymbol{x}\} \cup \{\boldsymbol{y}\}, W, \boldsymbol{r}) \\
&\geq R_2(A \cup \{\boldsymbol{x}\}, W, \boldsymbol{r}) + R_2(A \cup \{\boldsymbol{y}\}, W, \boldsymbol{r}).
\end{aligned} \tag{3}$$

By the definition of the R2 indicator with weighted Tchebycheff function in Definition 8, we know that

$$R_2(A, W, \boldsymbol{r}) = \frac{1}{|W|} \sum_{\boldsymbol{w} \in W} \min_{\boldsymbol{a} \in A} Tch(\boldsymbol{a}, \boldsymbol{w}, \boldsymbol{r}),$$

where $Tch(\boldsymbol{a}, \boldsymbol{w}, \boldsymbol{r}) = \max_{i\in\{1,\ldots,m\}}\{w_i|r_i - a_i|\}$. Thus, Eq. (3) is equivalent to

$$\begin{aligned}&\sum_{\boldsymbol{w}\in W}\min_{\boldsymbol{a}\in A\cup\{\boldsymbol{x}\}\cup\{\boldsymbol{y}\}}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}) + \sum_{\boldsymbol{w}\in W}\min_{\boldsymbol{a}\in A}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r})\\ \geq &\sum_{\boldsymbol{w}\in W}\min_{\boldsymbol{a}\in A\cup\{\boldsymbol{x}\}}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}) + \sum_{\boldsymbol{w}\in W}\min_{\boldsymbol{a}\in A\cup\{\boldsymbol{y}\}}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}).\end{aligned}$$

In fact, we can prove a stronger conclusion that for each $\boldsymbol{w} \in W$, it holds

$$\begin{aligned}&\min_{\boldsymbol{a}\in A\cup\{\boldsymbol{x}\}\cup\{\boldsymbol{y}\}}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}) + \min_{\boldsymbol{a}\in A}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r})\\ \geq &\min_{\boldsymbol{a}\in A\cup\{\boldsymbol{x}\}}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}) + \min_{\boldsymbol{a}\in A\cup\{\boldsymbol{y}\}}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}).\end{aligned} \tag{4}$$

For notational convenience, let

$$\begin{aligned}T_A &= \min_{\boldsymbol{a}\in A}Tch(\boldsymbol{a},\boldsymbol{w},\boldsymbol{r}),\\ T_{\boldsymbol{x}} &= Tch(\boldsymbol{x},\boldsymbol{w},\boldsymbol{r}) = \max_{i\in\{1,\ldots,m\}}\{w_i|x_i - r_i|\},\\ T_{\boldsymbol{y}} &= Tch(\boldsymbol{y},\boldsymbol{w},\boldsymbol{r}) = \max_{i\in\{1,\ldots,m\}}\{w_i|y_i - r_i|\}.\end{aligned}$$

Then Eq. (4) is equivalent to

$$\begin{aligned}&\min\{T_A, T_{\boldsymbol{x}}, T_{\boldsymbol{y}}\} + T_A\\ &\geq \min\{T_A, T_{\boldsymbol{x}}\} + \min\{T_A, T_{\boldsymbol{y}}\}.\end{aligned} \tag{5}$$

Without loss of generality, we suppose that $T_{\boldsymbol{x}} \leq T_{\boldsymbol{y}}$. Then, we can divide all possible relationships among T_A, $T_{\boldsymbol{x}}$ and $T_{\boldsymbol{y}}$ into three cases.

(1) $T_A \leq T_{\boldsymbol{x}} \leq T_{\boldsymbol{y}}$. We have $\min\{T_A, T_{\boldsymbol{x}}, T_{\boldsymbol{y}}\} + T_A = 2T_A = \min\{T_A, T_{\boldsymbol{x}}\} + \min\{T_A, T_{\boldsymbol{y}}\}$.
(2) $T_{\boldsymbol{x}} \leq T_A \leq T_{\boldsymbol{y}}$. We have $\min\{T_A, T_{\boldsymbol{x}}, T_{\boldsymbol{y}}\} + T_A = T_{\boldsymbol{x}} + T_A = \min\{T_A, T_{\boldsymbol{x}}\} + \min\{T_A, T_{\boldsymbol{y}}\}$.
(3) $T_{\boldsymbol{x}} \leq T_{\boldsymbol{y}} \leq T_A$. We have $\min\{T_A, T_{\boldsymbol{x}}, T_{\boldsymbol{y}}\} + T_A = T_{\boldsymbol{x}} + T_A \geq T_{\boldsymbol{x}} + T_{\boldsymbol{y}} = \min\{T_A, T_{\boldsymbol{x}}\} + \min\{T_A, T_{\boldsymbol{y}}\}$.

Thus, Eq. (5) holds, implying that the minus R2 indicator is submodular. ■

In [22, 26, 36], it has been proved that for the general subset selection problem, the GSEMO algorithm can achieve an approximation ratio of $(1-1/e)$ by running at most $2ek^2n$ iterations in expectation, when the objective function is monotone and submodular. Because the hypervolume and minus IGD/IGD+/R2 indicators are all monotone submodular, we directly have Theorem 3. That is, for each of the indicators, our corresponding algorithm outputs a subset with the indicator value at least $(1 - 1/e)$ times the optimal indicator value, by running at most $2ek^2n$ expected number of iterations. Note that the achieved $(1 - 1/e)$-approximation guarantee is the best-known (actually optimal) one, reaching that of the greedy algorithms in [16, 19].

Theorem 3. *For the subset selection problem in EMO using hypervolume, IGD, IGD+ or R2 indicator, our corresponding algorithm (i.e., GSEMO-HV, GSEMO-ACC-IGD, GSEMO-ACC-IGD+ or GSEMO-ACC-R2) achieves an* $(1 - 1/e)$*-approximation ratio by using at most* $2ek^2n$ *expected number of iterations, i.e.,* $\mathbb{E}[T] \leq 2ek^2n$*, where* k *and* n *are the budget of the size constraint* $|S| \leq k$ *and the size of the ground set* G *in Definition 4, respectively.*

V. Experiments

In this section, we introduce details of all our experiments. To begin with, we carry out experiments on benchmark as well as real-world datasets to demonstrate the general performance of our algorithms, i.e., our algorithms generally perform better than the greedy algorithms [16, 19] in practice. Moreover, we explore the scalability of our algorithms by using data sets with different sizes n and considering different budgets k. At last we compare popular MOEAs (i.e., NSGA-II and MOEA/D) with our algorithms.

A. Experiment data sets

Firstly, we use a benchmark dataset for the evolutionary multi-objective optimization subset selection problem [7]. The benchmark dataset involves a benchmark test suite for subset selection composed by two parts, which are sampled from the Pareto fronts and generated by EMO algorithms, respectively.

- The first part of benchmark dataset are candidate solution sets created by directly sampling points on different Pareto fronts such as the linear, convex, and concave Pareto fronts with triangular and inverted triangular shapes. For each candidate solution set, there are 10k solutions [7], and we randomly sample 1k solutions as our dataset. To make the experiments more comprehensive, we also generate a dataset from the Pareto front of the DTLZ7 problem with a disconnected Pareto front shape. Specifically, we use platEMO (a widely used open-source evolutionary algorithms and problems framework) [39] to generate 50k points uniformly distributed on the Pareto front of the DTLZ7 problem and then randomly select 1k points from them as our dataset. For the simplicity, this part of benchmark dataset, which is sampled directly from the Pareto fronts, is called BM1, and the shapes of Pareto fronts used in the BM1 dataset are provided in the supplementary material due to space limitation.
- The second part of this benchmark dataset are non-dominated solution sets generated by running the representative MOEAs (e.g., NSGA-II and MOEA/D) with the standard setting of the population size on some classic test problems, i.e., DTLZ1, DTLZ2, MinusDTLZ1, MinusDTLZ2, DTLZ7 and WFG3, after 10k solution evaluations. For details, the reader can refer to [7], and we name this dataset BM2.

We also use a real-world dataset [40], which is a non-dominated solution set obtained by NSGA-II on a 9-objective software test suite generation instance, Drupal [41], in software product line. In this 9-objective optimization problem, there are seven objectives to maximize and the remaining two objective functions to minimize. We use the candidate solutions generated by NSGA-II with the population size 100 after 100 generations, as practiced in [40]. We choose the non-dominated solutions from all these candidates as our dataset, which has a size of 429 solutions. To formalize this dataset, we transfer all maximization problems into corresponding minimization problems for the convenience and normalize our dataset by minimax scaler. More specifically, for a minimization problem, we normalize an objective value $f(\boldsymbol{x})$ by

$\frac{f(\boldsymbol{x})-minValue}{maxValue-minValue}$, and for a maximization problem, we normalize and convert it into a minimization problem by $\frac{maxValue-f(\boldsymbol{x})}{maxValue-minValue}$.

B. Experiment Settings

First of all, the number of solutions to be selected is specified as 10, i.e., the budget k in Definition 4 is set to 10, since users usually may not want to make choices from too many options. We further give our parameter settings for each indicator and some explanations for it.

1) IGD/IGD+ Subset Selection: Since a large reference set is preferable in the calculation of IGD [10], we use the whole candidate solution set as the reference set.

2) Hypervolume Subset Selection: In the hypervolume subset selection algorithm, the reference point for the hypervolume indicator is set as 1.1 times of the maximum on all objectives. Nevertheless, the calculation of the hypervolume value can be time-consuming. Thus, we use a variant of R2 indicator [42] to approximate hypervolume for acceleration. To make it fair, we still evaluate the final output using the hypervolume indicator. Specifically, we use the approximate hypervolume calculation in the subset selection process to accelerate the process. For the final output evaluation, we consider the exact hypervolume calculation. As regards to the direction vectors of the approximation indicator, we generate them by following the previous work [42]. More concretely, we use Das and Dennis's systematic approach [43], i.e., the simplex lattice design method [5], to generate a set of weight vectors and then for each weight vector $\boldsymbol{w}$ we get the corresponding direction vector $\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|_2}$. We set the number of directions to 1035, 1001, 1122, 1296 and 1430 for 3, 5, 8, 9 and 10-objective problems, respectively, which are decided by a combinatorial formula.

3) R2 Subset Selection: We set the Utopian point to be -0.1 times the maximum on each objective. According to the definition of R2 with the weighted Tchebycheff functions, we use Das and Dennis's systematic approach [43] to get a set of weight vectors $\boldsymbol{w}$ which strictly follow uniform distribution satisfying that $\|\boldsymbol{w}\|_1 = 1$. In our experiments, the number of weight vectors is set as 105, 105, 120, 165 and 220 for 3, 5, 8, 9 and 10-objective problems, respectively. These numbers are also decided by a combinatorial formula. The weight vectors in the 3-objective R2 subset selection problem are illustrated in the supplementary material due to space limitation.

For each test problem instance, we run two kinds of algorithms: the greedy algorithm [19, 42] as the baseline algorithm, and our algorithms (GSEMO or GSEMO-ACC) for each quality indicator. We set the number of iterations in our algorithms to be $2ek^2n$ so as to keep consistent with the theoretical analysis. As our algorithms are randomized, their run is repeatedly 30 times independently, and we report the average results and standard deviation. Note that the greedy algorithm is deterministic, and thus is run only once.

C. Results on Benchmark and Real-world Problems

We present our experimental results on the benchmark dataset (i.e., BM1 and BM2) and real-world dataset (i.e., Drupal) in Tables I to III. Since results on the benchmark dataset are enormous, we just show the summary of statistical significance test results in Tables I, II and III, and the detailed results are provided in the supplementary material.

Table I shows our experimental results on the BM1 dataset. We can observe that our algorithms (GSEMO and GSEMO-ACC) are generally better than the greedy algorithm. As for the IGD, IGD+ and R2 subset selection problems, GSEMO-ACC is significantly better than the greedy algorithm on most of the problem instances. On the hypervolume subset selection problem, though GSEMO is generally better than the greedy algorithm, there are a few cases in which the greedy algorithm seems better. This may be because we use the approximation of hypervolume (i.e., a variant of R2 indicator [42]) to accelerate, which may bring errors as mentioned in [42].

Besides, Table II shows our experimental results on the BM2 dataset, which are consistent with the results in Table I except that GSEMO seems not much better than the greedy algorithm for the hypervolume subset selection problem. In Table III, we display experimental results on the real-world dataset. We can find that our algorithms have a better performance than the greedy algorithm on the IGD, IGD+ and R2 indicators and return the same results as the greedy algorithm on the hypervolume indicator. Generally speaking, these experimental results in Tables I to III show that our algorithms can perform well on various datasets. The reason why the superiority of GSEMO is not so evident on the hypervolume indicator will be analyzed in Section V-E.

In general, our algorithms achieve better results at a cost of computational time. Take the 3-objective concave inverted triangular dataset in BM1 and the 3-objective minus DTLZ2 dataset in BM2 as two examples. For the former dataset, the computational time of GSEMO is about 267 times, 188 times, 49 times and 138 times longer than that of the lazy greedy algorithms for IGD, IGD+, hypervolume and R2 on average, respectively. For the latter dataset, the computational time of GSEMO is about 274 times, 172 times, 51 times and 149 times longer than that of the lazy greedy algorithms for IGD, IGD+, hypervolume and R2 on average, respectively. In this comparison, we use the lazy greedy algorithms proposed in Chen's work [16], which are dozens of times faster than the corresponding greedy algorithms. For the detailed computational time, please refer to the captions of Figs. 2 and 3.

TABLE I
SUMMARY OF STATISTICAL SIGNIFICANCE TEST RESULTS ON THE BM1 DATASET. THE THREE NUMBERS IN A CELL RESPECTIVELY INDICATE THE NUMBER OF PROBLEMS THAT: GREEDY IS SIGNIFICANTLY BETTER / ALMOST EQUIVALENT TO / INFERIOR TO GSEMO OR GSEMO-ACC ACCORDING TO THE WILCOXON RANK-SUM TEST WITH CONFIDENCE LEVEL 0.95

#Obj.	IGD	IGD+	Hypervolume	R2
3	0/0/7	0/0/7	3/0/4	0/0/7
5	0/0/7	1/0/6	1/5/1	0/0/7
8	0/1/6	0/0/7	1/1/5	0/2/5
10	0/0/7	0/2/5	0/5/2	0/3/4

TABLE II
SUMMARY OF STATISTICAL SIGNIFICANCE TEST RESULTS ON THE BM2 DATASET. THE TABLE SETTINGS IS THE SAME AS TABLE I.

#Obj.	IGD	IGD+	Hypervolume	R2
3	0/0/6	0/1/5	1/1/4	0/0/6
5	0/0/6	0/0/6	3/1/2	0/0/6
8	0/3/3	2/0/4	1/3/2	0/1/5
10	1/0/5	1/0/5	4/0/2	0/2/4

TABLE III
SUMMARY OF STATISTICAL SIGNIFICANCE TEST RESULTS ON THE REAL-WORLD PROBLEM: DRUPAL. THE TABLE SETTINGS IS THE SAME AS TABLE I.

IGD	IGD+	Hypervolume	R2
0/0/1	0/0/1	0/1/0	0/0/1

D. Scalability

Since real-world problems often have different scales, this part of experiments is to show that our algorithms can be generally good to handle problems with different sizes, demonstrating the good scalability of our algorithms.

We carry out experiments on the 3-objective dataset of the concave inverted triangular and disconnected Pareto front in the BM1 dataset and vary the size k of the solution set. More concretely, our experiments are to select 10, 15, 20, 25 and 30 solutions from 1k candidate solutions. Besides, we show our algorithms can behave well when the dataset gets larger through the experiment which is to select 10 solutions from 10k candidate solutions on the 3-objective dataset of the convex inverted triangular Pareto front in the BM1 dataset. The experimental results are shown in Table IV. Note that the total number of candidate solutions corresponds to the size of the ground set G in Definition 4, denoted as n in the table.

It is worth mentioning that our algorithms are more suitable for selecting a relatively small subset, since the theoretically computational time of GSEMO increases linearly with k^2 while the computational time of the greedy algorithm increases linearly with k. There is also some work on the hypervolume subset selection problem with small subsets [44].

E. Why Is The Greedy Algorithm Better Sometimes for Hypervolume Subset Selection?

As shown in the previous experimental results, although on a vast majority of the test problem instances our algorithm performs better than the greedy algorithm in terms of IGD, IGD+ and R2, this is not the case for hypervolume. The greedy algorithm performs better on many test instances for hypervolume, particularly when the number of objectives increases. In this section, we try to figure out why this happened.

Considering that GSEMO uses a variant of R2 indicator to approximate the hypervolume value, the reason why the greedy algorithm performs better is that during the execution of GSEMO the approximation indicator can be harder to approximate precisely and brings errors as the number of objectives increases. To verify our point of view, we compare the value of this approximate indicator of the solutions provided by the greedy algorithm and GSEMO, on all the experiments presented in Tables I to IV. The experimental results are provided in the supplementary material due to space limitation. These results prove our guess, which show that GSEMO does quite better than the greedy algorithm on this approximation indicator. It therefore reveals that the reason why GSEMO is not significantly better than the greedy algorithm on hypervolume is that the variant of R2 indicator may not precisely approximate the hypervolume indicator, which means that the employed number of direction vectors in the variant of R2 indicator is not accurate enough to reflect the hypervolume value.

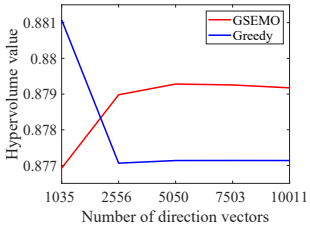

Fig. 1. Different levels of the hypervolume accuracy (by using different number of direction vectors) lead to different comparative results between GSEMO and the greedy algorithm on the 3-objective dataset of the linear triangular Pareto front in BM1.

We then empirically verify this conclusion by carrying out experiments on the 3-objective dataset of linear triangular Pareto front in the BM1 dataset, which varies the number of direction vectors of the approximation indicator to adjust the approximate accuracy to hypervolume. To be specific, we respectively set the number of direction vectors as 1035, 2556, 5050, 7503 and 10011. These experimental results shown in Fig. 1 reveal that GSEMO always behaves better than the greedy algorithm as the approximate accuracy promotes. In fact, the greedy algorithm performs worse as the approximate accuracy improves. This reveals that the unexpected excellent results returned by the greedy algorithm may be caused by the high randomness when the approximate accuracy is low. It matches our conclusion that GSEMO may be influenced by the error in the approximation of hypervolume indicator and demonstrates the superiority of this algorithm.

Additionally, it is desirable to study and improve the approximation quality on the R2-based hypervolume approximation. Interestingly, [45] recently reveals that Das and Dennis's systematic approach may not be the best method for the direction vector specification in the R2-based hypervolume contribution approximation.

F. What If Considering More Advanced MOEAs?

In the above experiments, we used the simple MOEA, GSEMO, which was typically used for theoretical study. And we found that our methods working with GSEMO perform significantly better than the greedy algorithm. Now one may ask what if we consider more advanced algorithms, like those commonly used in practice, for example, NSGA-II and MOEA/D. In this section, we do such an experiment on

TABLE IV
AVERAGE EXPERIMENT RESULTS ON THE SCALABILITY. THE BEST RESULT ON EACH PROBLEM INSTANCE IS BOLDED AND THE BULLET '•' DENOTES THAT THIS ALGORITHM IS SIGNIFICANTLY BETTER THAN THE CORRESPONDING ALGORITHM ACCORDING TO THE WILCOXON RANK-SUM TEST WITH CONFIDENCE LEVEL 0.95.

Problem	n	k	IGD		IGD+		Hypervolume		R2	
			Greedy	GSEMO-ACC	Greedy	GSEMO-ACC	Greedy	GSEMO	Greedy	GSEMO-ACC
3-objective Concave Inverted Triangular	1k	10	8.3861e-2	**8.2685e-2±5.6851e-4**•	3.7870e-2	**3.3613e-2±5.4836e-4**•	**5.6702e-2**	5.6584e-2±4.3538e-4	2.9393e-1	**2.9260e-1±9.6037e-5**•
		15	6.6780e-2	**6.5930e-2±4.1161e-4**•	2.6599e-2	**2.5492e-2±3.1375e-4**•	**6.2719e-2**	6.2712e-2±2.2239e-4	2.8980e-1	**2.8883e-1±9.0402e-5**•
		20	5.6705e-2	**5.4894e-2±4.7652e-4**•	2.1803e-2	**2.1399e-2±1.6882e-4**•	6.6135e-2	**6.6293e-2±3.8670e-4**	2.8733e-1	**2.8699e-1±6.0271e-5**•
		25	4.9083e-2	**4.7889e-2±3.0136e-4**•	1.8812e-2	**1.8547e-2±1.5501e-4**•	**6.9355e-2**•	6.9230e-2±1.9613e-4	2.8597e-1	**2.8577e-1±4.3978e-5**•
		30	4.3746e-2	**4.3148e-2±2.7732e-4**•	1.6692e-2	**1.6347e-2±1.3772e-4**•	7.1070e-2	**7.1092e-2±2.1134e-4**	2.8522e-1	**2.8505e-1±2.0475e-5**•
3-objective Disconnected	1k	10	1.7351e-1	**1.7388e-1±2.4251e-3**•	6.6854e-2	**6.4236e-2±7.6610e-4**•	1.3717	**1.4195±5.6512e-3**•	1.1051	**1.1048±2.2099e-5**•
		15	1.3540e-1	**1.3048e-1±1.1388e-3**•	5.2700e-2	**5.0693e-2±4.4173e-4**•	**1.4962**•	1.4938±3.5725e-3	1.1043	**1.1039±0**•
		20	1.1100e-1	**1.0785e-1±7.9486e-4**•	4.4356e-2	**4.2947e-2±2.7339e-4**•	1.5293	**1.5372±2.5965e-3**•	1.1039	**1.1038±4.5168e-16**•
		25	9.3397e-2	**9.2299e-2±5.6735e-4**•	3.8916e-2	**3.7827e-2±2.4809e-4**•	1.5584	**1.5669±2.3400e-3**•	1.1038	1.1038±4.5168e-16
		30	8.2178e-2	**8.2113e-2±6.5721e-4**•	3.5101e-2	**3.3730e-2±2.1706e-4**•	1.5871	**1.5873±2.2359e-3**•	1.1038	1.1038±4.5168e-16
3-objective Convex Inverted Triangular	10k	10	1.6206e-1	**1.5817e-1±1.4737e-3**•	**7.7369e-2**	7.7387e-2±4.7644e-4	5.7801e-1	**5.8217e-1±4.0167e-3**•	2.8694e-1	**2.8568e-1±7.2904e-5**•

the real-world dataset. Though these two MOEAs have no theoretical guarantee in the subset selection problem, they do quite well in many applications.

The experimental results are provided in the supplementary material due to space limitation and here we use Tables V to show the best algorithm on each indicator and the corresponding running time. Note that to ensure a fair comparison, we set the population size to 100 in NSGA-II and MOEA/D and the number of function evaluations to 233300 which is close to $2ek^2n$ of GSEMO. Generally, our GSEMO, NSGA-II and MOEA/D have better performance than the greedy algorithm, but spend much more time. In fact, the greedy algorithm only spends $2.1632e - 1/2.8278e - 1/1.9017e + 1/6.8950e - 1$ seconds on each indicator.

Another observation is that the more advanced algorithms NSGA-II and MOEA/D are not always better than GSEMO in this experiment. An explanation for that is that NSGA-II may be affected by keeping suboptimal solutions in its population for the reproduction procedure, while GSEMO always keeps non-dominated solutions in the population. MOEA/D behaves slightly inferiorly but is faster than NSGA-II. This may result from that MOEA/D uses uniformly distributed weight vectors to guide the search and for a relatively complex Pareto front, these weight vectors may not be able to guarantee the uniform distribution of candidate solutions [46]. Our observations may demonstrate the potential research directions on customizing algorithms for the subset selection problem in EMO.

G. Running Time

Considering the running time (in the number of objective function evaluations), Theorem 3 has shown that it requires at most $2ek^2n$ evaluations (note that one iteration of GSEMO costs one evaluation) in expectation for GSEMO to achieve the best-so-far approximation guarantee. We have used this theoretical upper bound in our experiments. Now we want to examine how efficient GSEMO can be in practice. By using two data sets from the BM1 and BM2 datasets respectively, we plot the curve of the average quality indicator values over the running time for GSEMO, and select the greedy algorithm as the baseline. To be concrete, we select the 3-objective dataset of the convex inverted triangular Pareto front in the BM1 dataset and the 3-objective dataset of the minus DTLZ2 in the BM2 dataset.

TABLE V
AVERAGE EXPERIMENTAL RESULTS OF THE GREEDY ALGORITHM, GSEMO/GSEMO-ACC, NSGA-II AND MOEA/D ON THE REAL-WORLD PROBLEM. HERE WE SHOW THE BEST ALGORITHM ON EACH INDICATOR AND THE CORRESPONDING RUNNING TIME.

Indicator	Best Algorithm	Result	CPU Time(s)
IGD	NSGA-II	7.7110e-2±2.8230e-17	3.7072e+1
IGD+	GSEMO-ACC	1.3251e-2±8.8219e-18	5.6742e+1
Hypervolume	NSGA-II	7.4972e-1±1.1292e-16	1.0027e+3
R2	GSEMO-ACC	9.7401e-2±2.6355e-7	1.0168e+2

We let our algorithms output the temporal solution set every kn iterations. The results are shown in Figs. 2 and 3. It can be seen that our algorithm only takes 10% of the theoretical time (i.e., $2ek^2n \approx 54kn$ where $k = 10$) to defeat the greedy algorithm. This confirms that our algorithm is even much more efficient practically compared to the theoretical results. This is expected, because the theoretical running time is the running time required by GSEMO to achieve a good approximation in the worst case.

VI. CONCLUSION

In this paper, we design a multi-objective evolutionary algorithmic framework to solve the subset selection problem in EMO. That is, the subset selection problem is reformulated as a bi-objective optimization problem (where the two objectives are quality indicator and subset size), and then solved by a MOEA. We propose instantiation of this general framework for representative quality indicators (i.e., hypervolume, IGD,

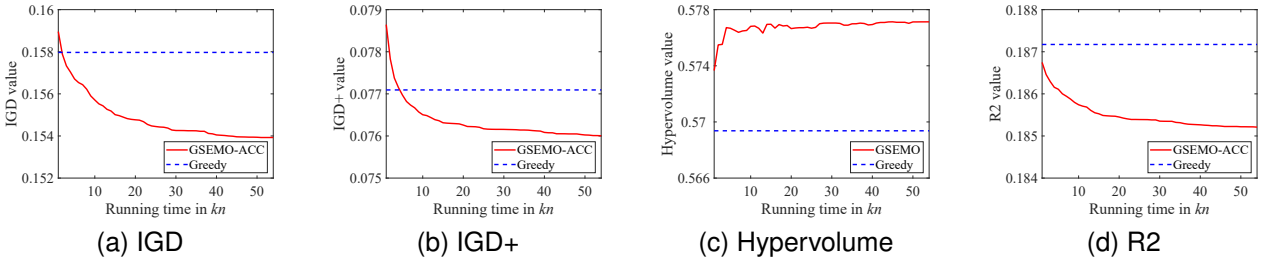

Fig. 2. Quality indicator value vs. running time of our algorithms (GSEMO and GSEMO-ACC) on the 3-objective convex inverted triangular Pareto front dataset in BM1, where the greedy algorithm is selected as the baseline. As for computational time, Greedy spent 5.2695e-1s, 1.0147s, 3.5575e+1s and 1.2867s, and GSEMO/GSEMO-ACC spent 1.4056e+2s, 2.0549e+2s, 1.7479e+3s and 1.7760e+2s in each subfigure, respectively.

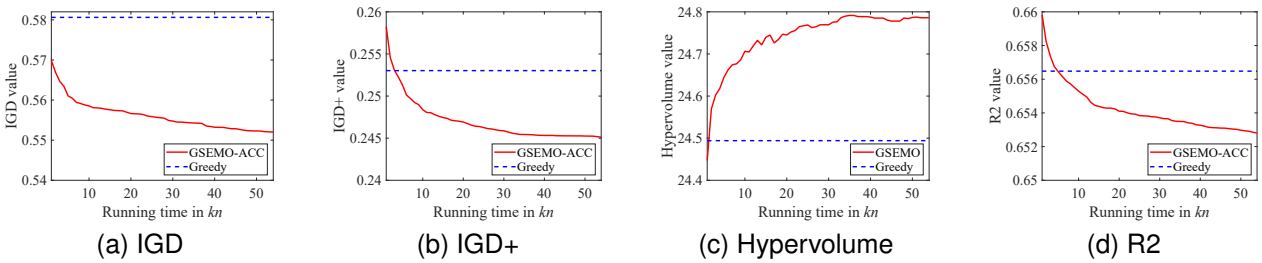

Fig. 3. Quality indicator value vs. running time of our algorithms (GSEMO and GSEMO-ACC) on the 3-objective minus DTLZ2 dataset in BM2, where the greedy algorithm is selected as the baseline. As for computational time, Greedy spent 1.1383s, 3.3438s, 6.5603e+1s and 2.5400s, and GSEMO/GSEMO-ACC spent 3.1244e+2s, 5.7420e+2s, 3.3714e+3s and 3.7899e+2s in each subfigure, respectively.

IGD+ and R2) based on GSEMO. Since GSEMO, despite performing well, is fairly slow for the problem in practice, we attempt an acceleration of the calculation of IGD, IGD+ and R2 indicators in GSEMO, and we also theoretically prove the acceleration ratio. Furthermore, we prove that our instantiated algorithms achieve the state-of-the-art approximation guarantee by utilizing the monotone submodular property of quality indicators. Through extensive experiments on benchmark and real-world datasets, we demonstrate promising performance of the proposed algorithms. Lastly, we empirically verify that well-established MOEAs (i.e., NSGA-II and MOEA/D) perform generally well, but may not improve the simple algorithm GSEMO much on this problem.

As regards to future work, it may be interesting to study on developing customized algorithms for the subset selection problem in EMO to further improve the performance. In addition, as presented in [47, 48] there may be some room to further accelerate our algorithms by parallel and distributed computing.

ACKNOWLEDGMENT

This work was supported by the National Science Foundation of China (62022039, 62276124). (Corresponding author: Chao Qian.)

REFERENCES

[1] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-objective Problems*. New York, NY: Springer, 2007.

[2] Z.-H. Zhou, Y. Yu, and C. Qian, *Evolutionary Learning: Advances in Theories and Algorithms*. Singapore: Springer, 2019.

[3] 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.

[4] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based non-dominated sorting approach, part II: Handling constraints and extending to an adaptive approach," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 602–622, 2013.

[5] 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.

[6] 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.

[7] K. Shang, T. Shu, H. Ishibuchi, Y. Nan, and L. M. Pang, "Benchmarking subset selection from large candidate solution sets in evolutionary multi-objective optimization," *Information Sciences*, vol. 622, pp. 755–770, 2023.

[8] H. Ishibuchi, L. M. Pang, and K. Shang, "A new framework of evolutionary multi-objective algorithms with an unbounded external archive," in *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI)*, Santiago, Spain, 2020, pp. 283–290.

[9] L. Paquete, B. Schulze, M. Stiglmayr, and A. C. Lourenço, "Computing representations using hypervolume scalarizations," *Computers & Operations Research*, vol. 137, p. 105349, 2022.

[10] M. Li and X. Yao, "Quality evaluation of solution sets in

multiobjective optimisation: A survey," *ACM Computing Surveys*, vol. 52, no. 2, pp. 1–38, 2019.

[11] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms — A comparative case study," in *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature (PPSN)*, Amsterdam, The Netherlands, 1998, pp. 292–301.

[12] C. A. C. Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *Proceedings of the 3rd Mexican International Conference on Artificial Intelligence (MICAI)*, Mexico City, Mexico, 2004, pp. 688–697.

[13] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *Proceedings of the 8th International Conference on Evolutionary Multi-criterion Optimization (EMO)*, Guimarães, Portugal, 2015, pp. 110–125.

[14] M. P. Hansen and A. Jaszkiewicz, *Evaluating the quality of approximations to the non-dominated set*. IMM, Department of Mathematical Modelling, Technical Universityof Denmark, 1994.

[15] G. Davis, S. Mallat, and M. Avellaneda, "Adaptive greedy approximations," *Constructive Approximation*, vol. 13, no. 1, pp. 57–98, 1997.

[16] W. Chen, H. Ishibuchi, and K. Shang, "Lazy greedy hypervolume subset selection from large candidate solution sets," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Glasgow, UK, 2020, pp. 1–8.

[17] M. Minoux, "Accelerated greedy algorithms for maximizing submodular set functions," *Optimization Techniques*, pp. 234–243, 1978.

[18] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher, "An analysis of approximations for maximizing submodular set functions-I," *Mathematical Programming*, vol. 14, no. 1, pp. 265–294, 1978.

[19] W. Chen, H. Ishibuchi, and K. Shang, "Fast greedy subset selection from large candidate solution sets in evolutionary multiobjective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 4, pp. 750–764, 2022.

[20] K. Shang, H. Ishibuchi, and W. Chen, "Greedy approximated hypervolume subset selection for many-objective optimization," in *Proceedings of the 23rd ACM Conference on Genetic and Evolutionary Computation (GECCO)*, Lille, France, 2021, pp. 448–456.

[21] H. Ishibuchi, L. M. Pang, and K. Shang, "Difficulties in fair performance comparison of multi-objective evolutionary algorithms," *IEEE Computational Intelligence Magazine*, vol. 17, no. 1, pp. 86–101, 2022.

[22] C. Qian, Y. Yu, and Z.-H. Zhou, "Subset selection by Pareto optimization," in *Advances in Neural Information Processing Systems 28 (NeurIPS)*, Montreal, Canada, 2015, pp. 1765–1773.

[23] 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.

[24] K. Bringmann, S. Cabello, and M. T. M. Emmerich, "Maximum volume subset selection for anchored boxes," in *Proceedings of the 33rd International Symposium on Computational Geometry (SoCG)*, Brisbane, Australia, 2017, pp. 22:1–22:15.

[25] T. Ulrich and L. Thiele, "Bounding the effectiveness of hypervolume-based (μ+λ)-archiving algorithms," in *Proceedings of the 6th International Conference on Learning and Intelligent Optimization (LION)*, Paris, France, 2012, pp. 235–249.

[26] C. Qian, Y. Yu, K. Tang, X. Yao, and Z.-H. Zhou, "Maximizing submodular or monotone approximately submodular functions by multi-objective evolutionary algorithms," *Artificial Intelligence*, vol. 275, pp. 279–294, 2019.

[27] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. Da Fonseca, "Performance assessment of multiobjective optimizers: An analysis and review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[28] J. D. Knowles, L. Thiele, and E. Zitzler, "A tutorial on the performance assessment of stochastic multiobjective optimizers," *TIK-report*, vol. 214, 2006.

[29] E. Zitzler, D. Brockhoff, and L. Thiele, "The hypervolume indicator revisited: On the design of Pareto-compliant indicators via weighted integration," in *Proceedings of the 4th International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, Matsushima, Japan, 2007, pp. 862–876.

[30] D. Brockhoff, T. Wagner, and H. Trautmann, "On the properties of the R2 indicator," in *Proceedings of the 14th ACM Conference on Genetic and Evolutionary Computation (GECCO)*, Philadelphia, PA, 2012, pp. 465–472.

[31] M. Li, T. Chen, and X. Yao, "How to evaluate solutions in Pareto-based search-based software engineering: A critical review and methodological guidance," *IEEE Transactions on Software Engineering*, vol. 48, no. 5, pp. 1771–1799, 2022.

[32] R. H. Gómez and C. A. C. Coello, "MOMBI: A new metaheuristic for many-objective optimization based on the R2 indicator," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Cancun, Mexico, 2013, pp. 2488–2495.

[33] D. H. Phan and J. Suzuki, "R2-IBEA: R2 indicator based evolutionary algorithm for multiobjective optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, Cancun, Mexico, 2013, pp. 1836–1845.

[34] T. Wagner, H. Trautmann, and D. Brockhoff, "Preference articulation by means of the R2 indicator," in *Proceedings of the 7th International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, Sheffield, UK, 2013, pp. 81–95.

[35] T. Friedrich, J. He, N. Hebbinghaus, F. Neumann, and C. Witt, "Approximating covering problems by randomized search heuristics using multi-objective models," *Evolutionary Computation*, vol. 18, no. 4, pp. 617–633,

2010.

[36] T. Friedrich and F. Neumann, "Maximizing submodular functions under matroid constraints by evolutionary algorithms," *Evolutionary Computation*, vol. 23, no. 4, pp. 543–558, 2015.

[37] C. Qian, "Multiobjective evolutionary algorithms are still good: Maximizing monotone approximately submodular minus modular functions," *Evolutionary Computation*, vol. 29, no. 4, pp. 463–490, 2021.

[38] A. P. Guerreiro, C. M. Fonseca, and L. Paquete, "The hypervolume indicator: Computational problems and algorithms," *ACM Computing Surveys*, vol. 54, no. 6, pp. 119:1–119:42, 2022.

[39] Y. Tian, R. Cheng, X. Zhang, and Y. Jin, "PlatEMO: A MATLAB platform for evolutionary multi-objective optimization," *IEEE Computational Intelligence Magazine*, vol. 12, no. 4, pp. 73–87, 2017.

[40] R. M. Hierons, M. Li, X. Liu, J. A. Parejo, S. Segura, and X. Yao, "Many-objective test suite generation for software product lines," *ACM Transactions on Software Engineering and Methodology*, vol. 29, no. 1, pp. 1–46, 2020.

[41] A. B. Sánchez, S. Segura, J. A. Parejo, and A. R. Cortés, "Variability testing in the wild: The drupal case study," *Software and Systems Modeling*, vol. 16, no. 1, pp. 173–194, 2017.

[42] K. Shang, H. Ishibuchi, M.-L. Zhang, and Y. Liu, "A new R2 indicator for better hypervolume approximation," in *Proceedings of the 20th ACM Conference on Genetic and Evolutionary Computation (GECCO)*, Kyoto, Japan, 2018, pp. 745–752.

[43] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM Journal on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[44] M. Basseur, B. Derbel, A. Goëffon, and A. Liefooghe, "Experiments on greedy and local search heuristics for *d*dimensional hypervolume subset selection," in *Proceedings of the 18th ACM Conference on Genetic and Evolutionary Computation (GECCO)*, Denver, CO, 2016, pp. 541–548.

[45] T. Shu, K. Shang, Y. Nan, and H. Ishibuchi, "Direction vector selection for R2-based hypervolume contribution approximation," in *Proceedings of the 17th International Conference on Parallel Problem Solving from Nature (PPSN)*, vol. 13399, 2022, pp. 110–123.

[46] M. Li and X. Yao, "What weights work for you? Adapting weights for any Pareto front shape in decomposition-based evolutionary multi-objective optimisation," *Evolutionary Computation*, vol. 28, no. 2, pp. 227–253, 2020.

[47] C. Qian, J.-C. Shi, Y. Yu, K. Tang, and Z.-H. Zhou, "Parallel Pareto optimization for subset selection," in *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, New York, NY, 2016, pp. 1939–1945.

[48] C. Qian, "Distributed Pareto optimization for large-scale noisy subset selection," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 4, pp. 694–707, 2020.

Yu-Ran Gu is a M.Sc. student in the School of Artificial Intelligence, Nanjing University, China. He received the B.Sc. degree from Department of Software Engineering in 2021 from Sun Yat-sen University. In September 2021, he was admitted to study for a M.Sc. degree in Nanjing University. His research interests are mainly designing efficient algorithms with theoretical guarantee for submodular optimization.

Chao Bian is a Ph.D. student in the School of Artificial Intelligence, Nanjing University, China. He received the B.Sc. degree from Department of Mathematics in 2016 from Nanjing University, and M.Sc. degree from School of Computer Science and Technology in 2019 from University of Science and Technology of China. In September 2020, he was admitted to study for a Ph.D. degree in Nanjing University. His research interests are mainly theoretical analysis of evolutionary algorithms (EAs) and design of efficient EAs under uncertain environments.

Miqing Li is an Assistant Professor at the University of Birmingham and a Turing Fellow of the Alan Turing Institute, UK. Miqing's research revolves around multi-objective optimisation, an area he is working on for 15 years. In general, his research consists of two parts: 1) basic research, namely, developing effective evolutionary algorithms for general challenging optimisation problems such as those with many objectives, complex constraints, numerous local/global optima, uncertainty, and expensive to evaluate, and 2) applied research, namely, designing customised search algorithms for practical problems in other fields such as those in software engineering, high-performance computing, neural architecture search, disassembly automation, and supply chain.

Chao Qian is an Associate Professor in the School of Artificial Intelligence, Nanjing University, China. He received the BSc and PhD degrees in the Department of Computer Science and Technology from Nanjing University. After finishing his PhD in 2015, he became an Associate Researcher in the School of Computer Science and Technology, University of Science and Technology of China, until 2019, when he returned to Nanjing University.

His research interests are mainly theoretical analysis of evolutionary algorithms (EAs), design of safe and efficient EAs, and evolutionary learning. He has published one book "Evolutionary Learning: Advances in Theories and Algorithms", and over 40 papers in top-tier journals (AIJ, ECJ, TEvC, Algorithmica, TCS) and conferences (AAAI, IJCAI, NeurIPS, ICLR). He has won the ACM GECCO 2011 Best Theory Paper Award, the IDEAL 2016 Best Paper Award, and the IEEE CEC 2021 Best Student Paper Award Nomination. He is an associate editor of IEEE Transactions on Evolutionary Computation, a young associate editor of Science China Information Sciences, an editorial board member of the Memetic Computing journal, and was a guest editor of Theoretical Computer Science. He is a member of IEEE Computational Intelligence Society (CIS) Evolutionary Computation Technical Committee, and was the chair of IEEE CIS Task Force on Theoretical Foundations of Bio-inspired Computation. He has regularly given tutorials and co-chaired special sessions at leading evolutionary computation conferences (CEC, GECCO, PPSN), and has been invited to give an Early Career Spotlight Talk "Towards Theoretically Grounded Evolutionary Learning" at IJCAI 2022.