Role of Sum of Objectives in Non-dominated Sorting to Reduce Dominance Comparisons

Sumit Mishra
Dept. of CSE
IIIT Guwahati
Guwahati, India
sumit@iiitg.ac.in

Aman Yadav
Dept. of CSE
IIIT Guwahati
Guwahati, India
aman.yadav121b@iiitg.ac.in

Akash Yadav
Dept. of CSE
Rajiv Gandhi Institute of Petroleum Technology
Amethi, India
akashy@rgipt.ac.in

***Abstract*—Non-dominated sorting is one of the important steps in Pareto dominance-based multiobjective evolutionary algorithms. In the last two decades, various approaches have been proposed to solve the non-dominated sorting problem. In non-dominated sorting the points are compared to obtain the dominance relationship between them and the points are assigned to their respective front based on the dominance relationship. Generally, the dominance relationship is obtained by comparing the objective values of the points. In this paper, we have used the sum of objectives for this purpose. We have experimentally used our dominance calculation approach in the existing approach and it has been noticed that using the sum of the objectives values helps in reducing the number of dominance comparisons.**

***Index Terms*—non-dominated sorting, dominance comparisons, objective value comparisons, sum of objectives**

I. Introduction

Pareto dominance-based multiobjective evolutionary algorithms (MOEAs) use various steps and non-dominated sorting is one of the important steps that rank the solutions based on the Pareto-dominance relation. This ranking is used by these algorithms for further processing. Non-dominated sorting is defined for the points that belong to a M-dimensional space $\mathbb{R}^M$. In MOEAs, each such M-dimensional point is an objective vector associated with a solution to a multi-objective problem. The set of such points is known as the population.

The approaches for non-dominated sorting do not consider any problem-related information and only focus on the objective vector associated with the solutions. Without loss of generality, we assume that all objectives, that correspond to the coordinates of the points, need to be minimized. Let there be a population $\mathbb{P} = \{P_1, P_2, \ldots, P_N\}$ in $\mathbb{R}^M$. A point $P_i \in \mathbb{P}$ is represented as $P_i = \langle P_{i_1}, P_{i_2}, \ldots, P_{i_M} \rangle$. The Pareto-dominance is defined as follows: a point $P_i = \langle P_{i_1}, P_{i_2}, \ldots, P_{i_M} \rangle$ dominates a point $P_j = \langle P_{j_1}, P_{j_2}, \ldots, P_{j_M} \rangle$, denoted as $P_i \prec P_j$, if both the following conditions are satisfied:

- $\forall m \in \{1, 2, \ldots, M\}\ P_{i_m} \leq P_{j_m}$
- $\exists m \in \{1, 2, \ldots, M\}\ P_{i_m} < P_{j_m}$.

In case point P_i does not dominate point P_j, it is denoted as $P_i \not\prec P_j$. When neither P_i dominates P_j nor P_j dominates P_i, *i.e.*, $(P_i \not\prec P_j) \wedge (P_j \not\prec P_i)$, then these two points P_i and P_j are said to be non-dominated. Non-dominated sorting divides the population $\mathbb{P}$ into different fronts $\{F_1, F_2, \ldots, F_K\}$ where $1 \leq K \leq N$, such that

- $\bigcup_{k \geq 1} F_k = \mathbb{P}$
- $\forall i \neq j,\ F_i \cap F_j = \emptyset$
- $\forall k \geq 1,\ \forall P_i, P_j \in F_k : (P_i \not\prec P_j) \wedge (P_j \not\prec P_i)$
- $\forall P_j \in F_1,\ \nexists P_i \in \mathbb{P} : P_i \prec P_j$;
- $\forall P_j \in F_k, k > 1\ \exists P_i \in F_{k-1} : P_i \prec P_j$.

A point $p \in F_k$ is said to have rank k. Assume the number of points in a front F_k be n_k, so $\sum_{k=1}^{K} n_k = N$. Let $P_i.\texttt{sum}$ denotes the sum of the objective values for point P_i, *i.e.*, $P_i.\texttt{sum} = P_{i_1} + P_{i_2} + \cdots + P_{i_M}$.

In this paper, we explore the sum of the objective values of all the points and use it for the computation of the dominance relationship between the points. This sum-based dominance relationship is then incorporated in one of the existing approaches and it has been shown that the sum-based dominance relationship helps to reduce the number of dominance comparisons.

In Section II, we summarize some of the existing non-dominated sorting approaches. In Section III, we propose an approach to compute the dominance relationship based on the sum of the objective values. In Section IV, we have evaluated the performance of the sum of the objectives-based dominance approach against a few existing approaches. Finally, Section V concludes the paper and gives the direction where the sum of the objectives-based dominance approach can be extended.

II. Related Work

In this section, we provide a summary of non-dominated sorting approaches. We can broadly divide the non-dominated sorting approaches into two categories – Sequential and Divide-and-conquer. One of the initial sequential approaches was proposed by Srinivas *et al.* [1] where a point can be compared with other points multiple times. The worst-case time complexity of this approach is $\mathcal{O}(MN^3)$. Fast Non-dominated sorting is proposed by Deb *et al.* [2] where all the points are compared with each other only once. This approach takes $\mathcal{O}(MN^2)$ time. Some of the approaches have been proposed based on the concept of presorting, *i.e.*, sorting the points based on either one of the objectives or based on all the objectives. This presorting has its benefit which is the points that come later in the sorted list cannot dominate the former points. Efficient Non-dominated Sort (ENS) [3] is one such approach that first sorts the points based on the first objective and then assigns the sorted points to their respective front. Two versions based on how the respective front is obtained have been proposed – sequential version (ENS-SS) and binary search-based version (ENS-BS). There are various other approaches [4], [5] that use the idea of pre-sorting. Gustavsson *et al.* [4] proposed an approach by extending the idea of ENS-BS [3] based on a Non-Dominated Tree (NDTree) which is a variant of a bucket k-dimensional tree (k-d tree). Zhang *et al.* [5] proposed an approach known as Tree-based Efficient Non-dominated Sorting (T-ENS) by extending the idea of ENS-SS [3]. This approach stores the points of a particular front in the form of $(M-1)$-ary tree. The worst-case time complexity of all these approaches is $\mathcal{O}(MN^2)$.

To reduce the number of floating-point comparisons in case the objective values are floating-point numbers, an approach based on the dominance degree of the point is presented in [6]. There have been some approaches like Best Order Sort [7], Bounded Best Order Sort [8] that sort the points based on all the objectives, unlike ENS [3] where the points are sorted based on a particular objective. Recently, an approach is known as Merge Non-dominated Sort (MNDS) [9] has been proposed which uses the concept of Java `bitset`.

There have been some approaches based on the divide-and-conquer strategy. Jensen [10] proposed an approach that takes $\mathcal{O}(N\log^{M-1}N)$. However, this approach is not suitable for the scenario when points share the same value for any objective. This limitation is removed by Fortin *et al.* [11] at the cost of an increased time complexity $\mathcal{O}(MN^2)$. A modified version of Fortin's algorithm is presented by Buzdalov *et al.* [12] with worst-case running time $\mathcal{O}(N\log^{M-1}N)$. Buzdalov [13] also proposed another approach using van Emde Boas tree by extending his work [12] with worst-case time complexity $\mathcal{O}(N\log^{M-2}N\log\log N)$.

III. Approach

In this section, we first discuss the role of the sum of objectives in computing the dominance relationship between two points.

Theorem 1. *Given a population $\mathbb{P} = \{P_1, P_2, \ldots, P_N\}$ in $\mathbb{R}^M$ that is sorted in ascending order based on the first objective. Given two points P_i and P_j in the sorted population where $j > i$. In this case,*

- *If $P_j.\texttt{sum} < P_i.\texttt{sum}$, then P_i and P_j are non-dominated, and this can be determined in $\mathcal{O}(1)$ time.*
- *Else, we need to obtain the dominance relationship by comparing the objective values of the points.*

Proof. Given a population $\mathbb{P} = \{P_1, P_2, \ldots, P_N\}$. When these points are sorted based on the first objective, then the points that come later in the sorted population cannot dominate the previous points [3]. Here, in case the value of the first objective is the same for two points, the value of the second objective is considered. If the value of the second objective is the same for two points, the value of the third objective is considered, and so on. In case the value of all the objectives is the same for two points, then any ordering of these points can be considered.

Let there be two points P_i and P_j where P_j comes later than P_i in the sorted population. As P_j comes later in the sorted population than P_i, it means for the first objective, the value of P_j is more than that of P_i. If P_i and P_j share the same value of the first objective then, the value of P_j is more than that of P_i for the second objective. If P_i and P_j share the same value of the second objective, then the value of P_j is more than that of P_i for the third objective, and so on. In general, the value of P_j is larger than P_i for r^{th} objective where both points have the same value for initial $r-1$ objectives.

When $P_j.\texttt{sum} < P_i.\texttt{sum}$, then for y^{th} objective $(y > r)$, the value of P_j is less than that of P_i. Hence for r^{th} objective, P_i is smaller and for y^{th} objective P_j is smaller hence both are *non-dominated*, so in this case there is no need to find the dominance relationship between P_i and P_j. In case $P_j.\texttt{sum} \not< P_i.\texttt{sum}$, then there can be two possible relationships – either P_i dominates P_j or P_i and P_j are non-dominated. □

The concept of the sum of objective values has been used in [14], where the authors have used the sum of objective values to sort the points initially. However, we are using it for dominance computation purpose. The sum-based dominance relationship discussed in Theorem 1 can be used in the approaches that use presorting of points before assigning rank to them. There are various approaches based on the concept of presorting like [3]–[5], [10]–[12], [15], [16], etc. The sum based dominance relationship approach is not suitable for the approaches [10]–[12] as these approaches divide the objective space. [15] is only for two objectives so we cannot save many objective value comparisons in dominance comparisons so we have not considered this approach. Improved Nondominated Sort [16] and ENS [3] work in the same manner. So we have considered ENS in our work. We can incorporate sum based dominance relationship approach in various other approaches proposed after ENS [3] like [4], [5], etc.

Algorithm 1 ENS FRAMEWORK

Input: $\mathbb{P}$: Population
Output: $\mathbb{F} = \{F_1, F_2, \ldots\}$: Points from $\mathbb{P}$ split into fronts

```
Sort the population ℙ in lexicographic order
𝔽 ← ∅                                   ▷ Initialize the set of fronts; initially empty
∀P_i ∈ ℙ, Compute P_i.sum = P_{i_1} + P_{i_2} + ··· + P_{i_M}
for each point χ ∈ ℙ do
    INSERT(𝔽, χ)
return 𝔽 = {F_1, F_2, ...}              ▷ Non-dominated fronts
```

ENS works in two phases – (i) Presorting and (ii) Assigning points to their respective front. In the first phase, the points are sorted based on their first objective value. The tie is resolved as discussed in Theorem 1. In the second phase, points are assigned to their respective front sequentially taken from the sorted list of points. The ENS framework is summarized in Algorithm 1. The first point is assigned to the first front without any dominance comparisons as no point dominates this point. Any subsequent point is assigned to a front F_k if and only if it is dominated by at least one point of all its previous fronts $F_1, F_2, \ldots, F_{k-1}$ and it is not dominated by all the previously assigned points of front F_k. The process to assign points to their respective front is summarized in Algorithm 2. In line no. 7 of Algorithm 2, we have used a method to compute the dominance relationship between χ and P and the method is summarized in Algorithm 3. This method compares the objective values of the points and decides their dominance relationship.

Algorithm 2 INSERT($\mathbb{F}, \chi$)

Input: $\mathbb{F} = \{F_1, F_2, \ldots\}$: Non-dominated fronts, χ: Point for insertion into set of fronts
Output: Updated set of fronts after insertion of χ
1: `isRanked` $\leftarrow$ FALSE $\triangleright$ χ is not yet ranked
2: **for each** front $F \in \mathbb{F}$ **do**
3: `isDominated` $\leftarrow$ FALSE
4: **for** each point $P \in F$ **do**
5: **if** χ.`sum` $<$ P.`sum` **then** $\triangleright$ χ is non-dominated with P
6: **else**
7: **if** DOMINANCE(P, χ) **then**
8: `isDominated` $\leftarrow$ TRUE $\triangleright$ χ is dominated
9: **BREAK** $\triangleright$ Check for next front
10: **if** `isDominated` $=$ FALSE **then**
11: $F \leftarrow F \cup \{\chi\}$ $\triangleright$ Add χ to F
12: `isRanked` $\leftarrow$ TRUE $\triangleright$ χ has been ranked
13: **BREAK**
14: **if** `isRanked` $=$ FALSE **then** $\triangleright$ χ has not been ranked
15: $F \leftarrow \emptyset$, $F \leftarrow F \cup \{\chi\}$, $\mathbb{F} \leftarrow \mathbb{F} \cup \{F\}$

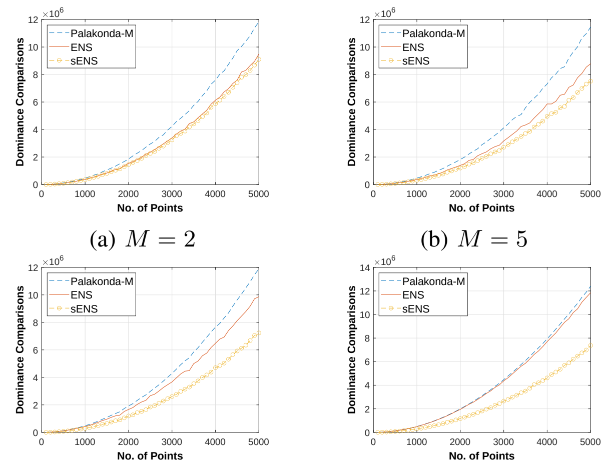
Algorithm 3 DOMINANCE(P_i, P_j)

Input: Points P_i and P_j where $j > i$
Output: TRUE: If $P_i \prec P_j$, FALSE: If P_i and P_j are non-dominated
`flag` $\leftarrow$ TRUE $\triangleright$ Points P_i and P_j are identical
for $m \leftarrow 1$ **to** M **do**
if $P_{j_m} < P_{i_m}$ **then** $\triangleright$ P_j is better than P_i on m-th obj
`flag` $\leftarrow$ FALSE $\triangleright$ Points are not identical
return FALSE $\triangleright$ P_i and P_j are non-dominated
else if $P_{j_m} > P_{i_m}$ **then** $\triangleright$ P_i is better than P_j on m-th obj
`flag` $\leftarrow$ FALSE $\triangleright$ Points are not identical
if `flag` $=$ TRUE **then**
return FALSE $\triangleright$ P_i and P_j are identical thus non-dominated
else
return TRUE $\triangleright$ P_i dominates P_j

The time complexity of ENS is $\mathcal{O}(MN^2)$. In our approach, we obtain the sum of objectives and then follow the procedure of ENS approach. As there are M objectives, the sum of a point will require $\Theta(M)$ time. Thus, the overall time to compute the sum is $\Theta(MN)$. So the time complexity of our approach is $\Theta(MN) + \mathcal{O}(MN^2)$ which is $\mathcal{O}(MN^2)$.

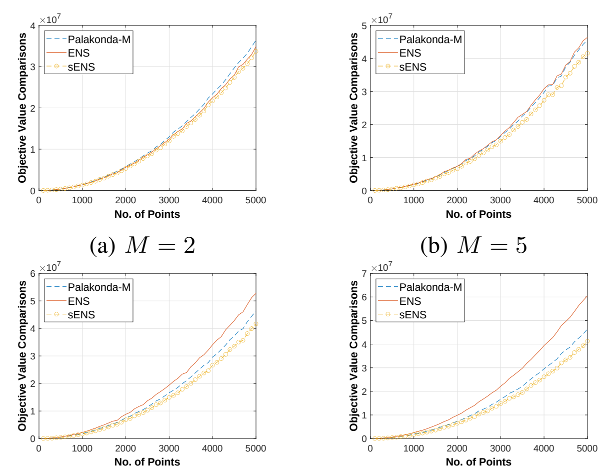
IV. EXPERIMENTAL EVALUATION

We have incorporated the sum-based dominance relationship in one of the existing approaches ENS [3]. We call our approach as `sENS` (Sum-based ENS). We have compared the performance of `sENS` with the ENS that uses a normal dominance relationship. We have also considered one more approach [14] for comparison which uses the sum of the objectives for sorting the points in the pre-sorting phase. We call this method `Palakonda-M` based on their author's name. We have used the cloud dataset for evaluating the performance as used by various approaches [3], [4], [7]–[9]. We have obtained the following information for all the three (`sENS`, `ENS` and `Palakonda-M`) approaches – (i) Number of dominance comparisons, *i.e.*, how many times the points are being compared, (ii) Number of objective value comparisons when the points are compared to obtain the dominance relationship and (iii) Execution time of the approaches.



(a) $M = 2$ (b) $M = 5$ (c) $M = 10$ (d) $M = 15$

Fig. 1: Performance of various approaches in terms of the number of dominance comparisons.

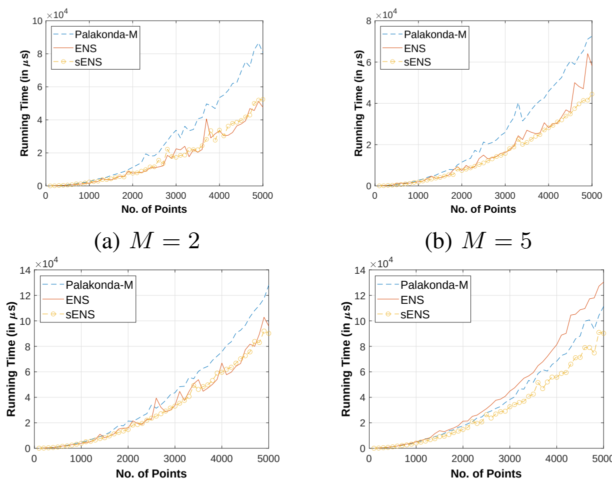


(a) $M = 2$ (b) $M = 5$ (c) $M = 10$ (d) $M = 15$

Fig. 2: Performance of various approaches in terms of the number of objective value comparisons.

In the cloud dataset, random points are generated from the interval $[0, 1]$ [7]. In this dataset, the number of fronts, and the cardinality of each front is random because of the randomly generated points. In our experiment, we have considered 50 populations with size ranging from 100 to 5000 with an increment of 100 as done in [3]. We have considered 2, 5, 10 and 15 as the number of objectives. The number of dominance comparisons is shown in Fig. 1. From this figure, it is clear that with an increase in the number of objectives, `sENS` takes less number of dominance comparisons than other approaches. A similar performance can be observed in the case of objective value comparisons which is shown in Fig. 2. The execution time of the approaches (in microseconds) is shown in Fig. 3. From this figure, we can observe that with an increase in the number of objectives, the performance of our approach improves as compared to others.

Discussion: From Fig. 1, we can see that the sum-based dominance relationship saves dominance comparisons However, it is possible to come up with a set of points in the population where we can guarantee that the number of dominance comparisons using sum-based dominance relationship and normal dominance relationship will be the same. To generate such a scenario, we have to make sure that the points that come later in the sorted list cannot have a smaller sum than the former points. Assume a population with N points. Let these points be equally divided into $N/2$ fronts where each front has two points. Such a population in $\mathbb{R}^2$ is shown in Table I.



(a) $M = 2$ (b) $M = 5$

(c) $M = 10$ (d) $M = 15$

Fig. 3: Execution time (in μs) of various approaches.

TABLE I: Scenario where sum-based dominance relationship and normal dominance relationship remains the same.

F_1	$\langle 1,2 \rangle, \langle 2,1 \rangle$
F_2	$\langle 3,4 \rangle, \langle 4,3 \rangle$
F_3	$\langle 5,6 \rangle, \langle 6,5 \rangle$
$\vdots$	$\vdots$
$F_{N/2-1}$	$\langle N-3, N-2 \rangle, \langle N-2, N-3 \rangle$
$F_{N/2}$	$\langle N-1, N \rangle, \langle N, N-1 \rangle$

V. Conclusion and Future Work

In this paper, we have explored a different way to obtain the dominance relationship between points using the sum of the objectives. The sum-based dominance relationship approach cannot always find the relation. However, it can help in many scenarios. Also, in the scenario where the sum-based dominance relationship approach cannot help, the number of dominance comparisons will be the same as in the normal relationship approach. Thus, our approach cannot be inferior to the normal dominance relationship approach if not superior. To validate our approach, we have used ENS [3]. In the future, we can evaluate the performance of our approach considering various non-dominated sorting approaches that use presorting.

References

[1] N. Srinivas and K. Deb, "Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, Fall 1994.

[2] 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, April 2002.

[3] X. Zhang, Y. Tian, R. Cheng, and J. Yaochu, "An Efficient Approach to Nondominated Sorting for Evolutionary Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 2, pp. 201–213, April 2015.

[4] P. Gustavsson and A. Syberfeldt, "A New Algorithm Using the Non-Dominated Tree to Improve Non-Dominated Sorting," *Evolutionary Computation*, vol. 26, no. 1, pp. 89–116, September 2018.

[5] X. Zhang, Y. Tian, R. Cheng, and Y. Jin, "A Decision Variable Clustering-Based Evolutionary Algorithm for Large-Scale Many-Objective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 1, pp. 97–112, February 2018.

[6] Y. Zhou, Z. Chen, and J. Zhang, "Ranking Vectors by Means of the Dominance Degree Matrix," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 34–51, 2017.

[7] P. C. Roy, M. M. Islam, and K. Deb, "Best Order Sort: A New Algorithm to Non-Dominated Sorting for Evolutionary Multi-Objective Optimization," July 20-24 2016, pp. 1113–1120.

[8] P. C. Roy, K. Deb, and M. M. Islam, "An Efficient Nondominated Sorting Algorithm for Large Number of Fronts," *IEEE Transactions on Cybernetics*, no. 99, pp. 1–11, 2018.

[9] J. Moreno, D. Rodriguez, A. J. Nebro, and J. A. Lozano, "Merge Nondominated Sorting Algorithm for Many-objective Optimization," *IEEE Transactions on Cybernetics*, vol. 51, no. 12, pp. 6154–6164, 2020.

[10] M. T. Jensen, "Reducing the Run-Time Complexity of Multiobjective EAs: The NSGA-II and Other Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 5, pp. 503–515, October 2003.

[11] F.-A. Fortin, S. Greiner, and M. Parizeau, "Generalizing the Improved Run-Time Complexity Algorithm for Non-Dominated Sorting," in *Proceedings of Genetic and Evolutionary Computation Conference (GECCO'2013)*, July 2013, pp. 615–622.

[12] M. Buzdalov and A. Shalyto, "A Provably Asymptotically Fast Version of the Generalized Jensen Algorithm for Non-Dominated Sorting," in *International Conference on Parallel Problem Solving from Nature – PPSN XIII*, September 13-17 2014, pp. 528–537.

[13] M. Buzdalov, "Make Evolutionary Multiobjective Algorithms Scale Better with Advanced Data Structures: Van Emde Boas Tree for Non-Dominated Sorting," in *International Conference on Evolutionary Multi-Criterion Optimization (EMO'2019)*, March 10-13 2019, pp. 66–77.

[14] V. Palakonda, T. Pamulapati, R. Mallipeddi, P. P. Biswas, and K. C. Veluvolu, "Nondominated Sorting Based on Sum of Objectives," in *IEEE Symposium Series on Computational Intelligence (SSCI'2017)*. IEEE, 2017, pp. 1–8.

[15] M. Liu and W. Zeng, "Reducing the Run-time Complexity of NSGA-II for Bi-objective Optimization Problem," in *International Conference on Intelligent Computing and Intelligent Systems (ICIS'2010)*, vol. 2. IEEE, 2010, pp. 546–549.

[16] A. R. da Cruz, "An Improved Nondominated Sorting Algorithm," *International Journal of Natural Computing Research*, vol. 3, no. 4, pp. 20–42, 2012.