Croatian Operational Research Review
CRORR **17:1**(2026), 111–123

A hybrid approach to approximate the Pareto front of the MOST problem

Asma Boumesbah[1,*] **and Mohamed El-Amine Chergui**[1]

[1] *Faculty of Mathematics, University of Science and Technology Houari Boumediene, RECITS Laboratory, BP 32, El Alia, Bab Ezzouar, Algiers, Algeria.*
E-mail: ⟨{aboumesbah,mchergui}@usthb.dz⟩

Abstract. This study introduces a hybrid NSGA-II algorithm with Multi-VNS for approximating the Pareto front of the multi-objective spanning tree (MOST) problem, building on recent approaches that have adapted NSGA-II combined with local search heuristics. By exploiting the property that a spanning tree is acyclic and that the addition of an edge generates a unique cycle, our mutation operator adds edges to a given spanning tree T of a connected graph G, thereby reducing the size of the MOST problem. By applying the exact mutation operator with low probability, this reduced problem is solved, producing a set of mutant solutions. The NSGA-II selection operator then approximates the Pareto front, which is further refined by a Multi-VNS metaheuristic to balance diversification and intensification. Comparative experiments with both exact and approximate methods demonstrate promising results.

Keywords: hybridization methods, metaheuristics, multi-objective spanning tree problem

Received: November 5, 2024; accepted: September 10, 2025; available online: December 9, 2025

DOI: 10.17535/crorr.2026.0009

Original scientific paper.

1. Introduction

Multi-objective optimization problems aim to identify efficient solutions that consider multiple, often conflicting objectives. One prominent example is the Multi-Objective minimum Spanning Tree (MOST), which is applied in various systems like communication networks, electric power systems, drainage systems, physical systems design, reducing data storage, cluster analysis. The authors in [17] applied a MOST approach under uncertainty conditions to optimize the distribution of petroleum products.
Several studies highlight the effectiveness of evolutionary and hybrid metaheuristics in approximating Pareto fronts for complex multi-objective problems. For example, [4] shows how advanced evolutionary algorithms can capture trade-offs among conflicting objectives using real-world data, underscoring the relevance of hybrid strategies for the MOST problem.
The MOST problem is well known to be NP-hard [5], making exact methods impractical for large instances. Consequently, approximate methods have been developed to produce a Pareto front that closely approximates the exact front for the MOST problem, providing feasible solutions within reasonable time frames.
We consider an undirected connected graph where each edge has an associated cost-vector of dimension $r \geq 2$. A cost-vector Z of dimension r dominates another cost-vector W of the same dimension, if Z is at least as good as W in all dimensions and they are not equal. In this case,

*Corresponding author.

a spanning tree T of G is Pareto optimal if its vector weight is not dominated by any vector weight of another spanning tree of G. The image of the Pareto optimal set forms the Pareto front set.

The main objective of this paper is to present a method for generating non-dominated points for the MOST problem that can effectively approximate the Pareto front set. The proposed algorithm, which combines the strengths of NSGA-II and "Variable Neighborhood Search" (VNS) [15], is referred to as the HMOST-Algorithm. The algorithm utilizes a two-point crossover operator and a $k-opt$ procedure as a mutation operator with a low probability. Subsequently, the VNS algorithm is used to refine a subset of points on the Pareto front generated by NSGA-II. Computational experiments demonstrate that the HMOST-Algorithm outperformes previous methods in terms of speed, scalability in solving instances with complete graphs of more than 200 nodes, and its ability to discover both supported and non-supported spanning trees.

The rest of the paper is organized as follows. Section 1 provides the basic concepts and notations used throughout this work. The procedures of the proposed HMOST-Algorithm are reported in Section 2. Section 3 reports numerical experimentation performed on instances described in [20] and other randomly generated instances. Finally, Section 4 concludes the study.

2. Background concepts

Consider a simple connected and undirected graph $G = (V, E)$ of order n, $V = \{v_1, v_2, \ldots, v_n\}$ is a set of vertices and $E = \{e_1, e_2, \ldots, e_m\}$ is a set of edges. Each edge $e_i \in E$, with $i \in \{1, \ldots, m\}$, is valued by a cost-vector $c(e_i) = (c_{ik})$, $k \in \{1, \ldots, r\}$, $r \geq 2$.

Example: Consider a connected undirected graph with 6 vertices and 13 edges, where each edge is weighted according to three criteria. The mathematical model associated with the

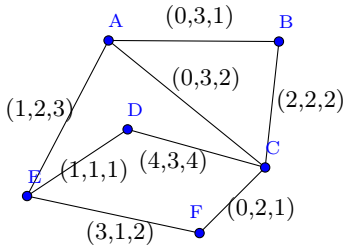

Figure 1: *A connected graph.*

MOST problem is expressed as follows:

$$(P)\begin{cases} Min\ C_k(x) = \sum_{i=1}^{m} c_{ik}x_i, & k \in \{1, \ldots, r\} \\ \sum_{i=1}^{m} x_i = n-1, & \\ \sum_{e_i \in E(S)} x_i \leq |S|-1, & \forall S \subset V, S \neq \emptyset \\ x_i \in \{0,1\}, & \forall j \in \{1, \ldots, m\}, \end{cases} \tag{1}$$

where $x = (x_i)_{i\in\{1,\ldots,m\}}$, $x_i = 1$ if the edge $e_i \in T$, and $x_i = 0$ otherwise. $E(S)$ is the set of edges of the subgraph induced by S, $S \subset V$.

Let $T \subset E$ a spanning tree of G. The cost-vector of T is given by $C_k(T) = \sum_{e_i \in T} c_{ik}, \quad k \in \{1, \ldots, r\}$. We denote $C(T) = (C_k(T))_{k \in \{1,\ldots,r\}}$.

- The vector $C(T)$ dominates another vector $C(T')$ if $C_k(T) \leq C_k(T')$ for all $k \in \{1, \ldots, r\}$, and $C_{k_0}(T) < C_{k_0}(T')$ for at least one index $k_0 \in \{1, ..., r\}$.

- T is efficient if there is no spanning tree T' of G such that $C(T')$ dominates $C(T)$.

- For any efficient spanning tree in the decision space, there is a corresponding non-dominated point in the criteria space. However, multiple efficient spanning trees can correspond to the same non-dominated point. The set of all efficient spanning trees is called the Pareto optimal set and the corresponding set of non-dominated points is called the Pareto front set. Let $\overline{T}$ the following set of edges: $\overline{T} = \{e \in E : e \notin T\}$. For each edge e belonging to $\overline{T}$, the set of edges $T \cup \{e\}$ contains a unique cycle μ_e and the set $B = \{\mu_e, e \in \overline{T}\}$ constitutes a cycle-basis of the graph G.

3. Related work

The multi-objective spanning tree problem has attracted much interest due to its practical applications in various fields. Researchers have developed numerous algorithms and heuristics to tackle this problem, ranging from exact methods like branch-and-bound and dynamic programming to approximate approaches such as genetic algorithms and multi-objective evolutionary algorithms. The challenge lies in balancing the trade-offs between different objectives to find a set of Pareto-optimal solutions that provide decision-makers with a range of trade-off choices. Furthermore, the complexity of the problem increases with the size of the graph and the number of objectives, making it a rich area for ongoing research and development.

In [22], an adaptation of the Genetic Algorithm (GA) is developed for the bi-objective minimum spanning tree problem using Prüfer number encoding [12], along with an early version of the Non-dominated Sorting Genetic Algorithm (NSGA) for giving out Pareto optimal spanning trees as closely as possible to the ideal point. To evaluate the performance of their adapted GA, the authors also proposed an algorithm intended to enumerate all Pareto-optimal spanning trees. However, [16] points out that this enumeration algorithm is incorrect and criticize it for failing to accurately compute the non-dominated solutions. They propose corrections and improvements, including a new enumeration method and an enhanced non-generational genetic algorithm for the MOST problem using a novel crossover operator (Dislocation Crossover) and a niche evolution strategy.

In [14] a novel genetic algorithm for the MOST problem is proposed while preserving the topoligical propreties of the graph. We note that algorithms [14, 16] have been tested on small graphs with two and three criteria. A particle swarm algorithm to solve problems with more than two objectives is presented in [13], but no experimental results are provided.

The authors in [8] introduce a Knowledge-based Evolutionary Algorithm (KEA), which demonstrates superior performance compared to the "Non-Sorting Genetic Algorithm" (NSGA-II) [9] algorithm in terms of both spread and the number of solutions identified. They initialize the population by leveraging extreme points to create an elite set of parents. Subsequently, the algorithm employs knowledge-based mutation exclusively to explore and uncover the remaining solutions on the Pareto front. A marking scheme is suggested to mitigate the reproduction of identical solutions and avoid dominated regions within the search space.

The study in [19] explores how local search techniques can enhance the overall effectiveness of NSGA-II for tackling the MOST problem, yet it lacks a comparative analysis against other existing methods in the literature. The authors evaluated the performance of three general-purpose local searches (Pareto Local Search, Tabu Search and Path Relinking) adapted to the multi-objective approach. Experimental results particularly with two and three objective functions, show that incorporating Pareto Local Search (PLS) in NSGA-II (NSGA-II+PLS) provides better performance.

To the best of our knowledge, these last two methods described above represent the most recent approximate approaches capable of handling more than two criteria and finding supported and non-supported Pareto solutions for large instances. This strongly motivated us to use them as references in our comparative study.

The survey presented in [11] reports the outcomes of a computational experiment featuring complete and grid graphs. It analyses the distinctive characteristics of each algorithm and evaluates the computational resources required to solve the instances.
Several exact methods have been proposed to solve the MOST problem. One such method, proposed in [3], generates all non-dominated points for problems involving at least two criteria. This branch-and-bound algorithm involves two main steps: (1) separating edges shared by at least two cycles, which allows the formulation of constraints as linear equalities, and (2) adding sub-cycle elimination inequalities to prevent cycles. Each branch of the search tree generates a multi-objective linear program with binary variables ($MOLBP$), representing the MOST problem at that branch.

The method in [6] targets only the supported solutions of the MOST problem. It begins by finding a lexicographical solution, then uses a neighborhood search starting from this optimal solution to identify the remaining supported spanning trees and their associated indifference regions. The study in [1] proposes a two-phase algorithm for the bi-objective minimum spanning tree problem. The first phase identifies extreme supported efficient solutions by combining mathematical programming with algorithmic techniques. The second phase identifies the remaining efficient solutions through a recursive edge interchange process, utilizing ascending non-zero reduced costs from weighted linear programs.

The method in [18] redefines the MOST problem as a One-to-One Multi-Objective Shortest Path (MOSP) problem, represented by a transition graph. The graph size is reduced using cost-based pruning criteria, and the Implicit Graph Multi-Objective Dijkstra algorithm is applied to solve the optimized MOSP instance, leveraging recent algorithmic advancements.

4. Description of the proposed method

One of the main reasons why population-based metaheuristics are favored over trajectory-based metaheuristics in multi-objective optimization is that they work with multiple solutions simultaneously, unlike traditional approaches that focus on a single solution. To address this disparity, we propose using the front obtained during the NSGA-II search process as the initial population for the VNS metaheuristic. The motivation for the hybrid algorithm is to combine the strengths of both methods to improve optimization performance.

Given the known convergence properties and limitations of NSGA-II, integrating the exact mutation operator, as well as the Multi-VNS framework, enhances the search dynamics. Specifically, this hybridization increases the algorithm's ability to improve the density of solutions along the obtained front and promotes more effective convergence toward the Pareto front. By combining these algorithms, the hybrid approach achieves a balance between exploration and exploitation, increases convergence speed, and improves the overall quality of solutions.

The proposed algorithm is outlined in the following procedures, describing the distinct steps of our algorithm, called the HMOST-Algorithm.

4.1. Encoding scheme

Initially, each edge in the graph $G = (V, E)$ consisting of n vertices and m edges, is assigned a unique identifier or number. The direct encoding of a chromosome is a set of $n - 1$ dimension, wherein each element represents the associated number of an edge in the corresponding spanning tree. Prüfer coding is an encoding method using a sequence of $n - 2$ natural numbers for a

spanning tree as proposed in [22]. The authors in [16], point out that the direct encoding is better and more efficient than Purfer-based encoding.

4.2. Starting population

The initial population, comprising s spanning trees, is generated through:
1- The optimal spanning trees corresponding to each criterion,
2- Random generation with weighted-sum method,
3- Applying Kruskal's algorithm by randomly selecting edges of the graph G.

It is worth noting that the first two procedures exclusively produce supported solutions, contrasting with the third, which specifically seeks non-supported solutions. Furthermore, this approach consistently ensures the attainment of feasible solutions and a well-suited equilibrium between individual quality and diversity.

4.3. Crossover operator

The reproduction process involves combining two individuals through the crossover operator. This is achieved by randomly generating a number, denoted as ρ, within the interval [0,1]. Crossover occurs only if $\rho \leq 0.80$. In such instances, two-point crossover is employed:

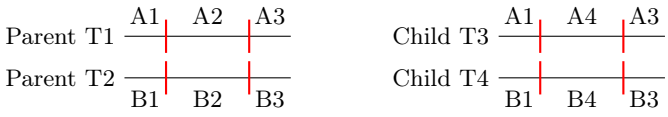

Figure 2: *2-point crossover.*

Two cut-points are randomly generated. The obtained offspring chromosomes do not correspond necessarily to spanning trees. To overcome this infeasibility, procedures of rearrangement of edges are introduced.

The two-point crossover is explained as follows, based on Figure 2:
Select edges of $A4$ in ascending order of the sums of costs in the set: $T2 \cup E \setminus (A1 \cup A3)$. Those of $B4$ are chosen in ascending order of the sums of costs in the set: $T1 \cup E \setminus (B1 \cup B3)$.

4.4. Diversified mutation operator

The mutation is achieved as in the biological evolutionary process; its priority aim is to generate better solutions in the sense of dominance. The diversity aspect is still present even in the case of non-improvement. The mutation operator is applied to the offsprings of the current population with a low predetermined probability $pm = 0.20$. The task involves creating a partial graph H from an offspring T by adding k edges from $E \setminus T$, $k \in \{1, 2, 3\}$. Hence, H contains k cycles. We then consider the subgraph L of H formed by only these cycles. The process of generating mutated offspring by adding k edges is called k-opt mutation.

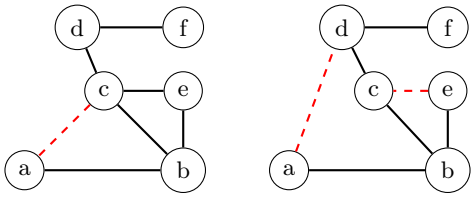

Figure 3: *Illustration of 1-opt (left): one edge added to form a single cycle L; and 2-opt (right): two edges added to form two cycles building the subgraph L. Red dashed edges are candidates added during mutation.*

Applying the 1-opt mutation, the subgraph L is reduced to an elementary cycle μ. If μ contains a dominated edge e, we obtain a spanning tree of L. This results in a spanning tree T' in H, $T' = E(H) \setminus \{e\}$, where $E(H)$ is the edge set of H. When we apply 2-opt mutation and 3-opt mutation, the subgraph L has a reduced dimension compared to the initial graph G. In these cases, we use our exact method, as described in [3], to generate all spanning trees in subgraph L corresponding to non-dominated points. As with 1-opt mutation, we reconstruct all non-dominated points for the partial graph H by sequentially adding the edges of $H \setminus L$ for each non-dominated points found previously. Unlike the 1-opt mutation, which is executed with high probability, the 2-opt mutation and 3-opt mutation are executed with low probability to minimize execution time.

Compared to standard heuristic mutations, this exact mutation operator ensures precision and can yield higher-quality offspring. However, due to computational cost, 2-opt and 3-opt are used less frequently.

Algorithm 1: k-opt Diversified Mutation

Spanning tree T, edge set E, parameter k the set of spanning trees T' $H \leftarrow T \cup$ random k edges from $E \setminus T$;
Identify subgraph L of H containing the k cycles;
$k = 1$ Find dominated edge e in the cycle;
$T' \leftarrow E(H) \setminus \{e\}$; Enumerate non-dominated trees in L using exact method;
non-dominated $T_L \subseteq L$ $T' \leftarrow T_L \cup (H \setminus L)$; the set of spanning trees T'

4.5. Dynamic selection operator

The selection operator starts by dividing the current population (with at least $2s$ individuals) into dominance fronts. The first front consists of non-dominated individuals, and subsequent fronts are formed by removing individuals from the previous front while maintaining non-dominance. The new population is created by selecting individuals according to the dominance order of the fronts. If additional individuals are needed to fill the population size s, individuals from front F_l ($l \geq 2$) are selected based on their "crowding" distance. If the first front is sufficient to fill the population, the size remains dynamic, determined by the first front's size.

4.6. Multi-VNS strategy

Using the set of spanning trees generated by the adapted NSGA-II algorithm, labeled MST, and the set ND representing its non-dominated points, a Multi-VNS-based algorithm generates neighboring solutions aimed at improving the set MST in order to obtain the set ND_{HMOST}, which contains non-dominated points for the HMOST-Algorithm. The ND_{HMOST} set is updated according to the Pareto dominance relation, and the populations obtained do not necessarily have the same size. The steps of the improvement by the Multi-VNS algorithm are described as follows:

- The set $P(MST)$ is a subset of the efficient solutions constructed by the NSGA-II algorithm, randomly chosen and having a cardinality of 10.
- The set N_k consists of neighborhood structures for $k = 1, 2, 3$. Given a spanning tree T (where $T \in P(MST)$), k edges are randomly selected from $E \setminus T$ and added to T. Consequently, k elementary cycles are created in the partial graph H, which is composed of the edges of T and the k added edges.

Each neighbor of T, according to N_k, can be reached by changing exactly k edges, with each edge belonging to a different elementary cycle of H.

Algorithm 2: Neighborhood Structure N_k

T: Spanning tree, E: Set of edges T': A neighbor of T
Randomly select k edges $e_1, e_2, ..., e_k$ from $E \setminus T$;
each selected edge e_i Find the elementary cycle $\mu(e_i)$ in $H = T \cup \{e_1, ..., e_k\}$;
Identify an edge $f \in T$ in the cycle $\mu(e_i)$;
Create $T' := T \setminus \{f\} \cup \{e_i\}$; T'

Algorithm 3: Multi-VNS Procedure

$P(MST)$: Set of spanning trees MST, ND_{HMOST}: Sets of efficient spanning trees and associated non-dominated points each solution T in $P(MST)$ $P(MST) := P(MST) \setminus \{T\}$;
a fixed number of iterations is not satisfied Randomly select a neighborhood structure N_k, with $k = 1, 2, 3$;
Generate a solution T' using N_k;
$C(T')$ dominates or equals $C(T)$ $T := T'$;
Update MST and ND with respect to non-dominance; MST and $ND_{HMOST} := ND$

Algorithm 4: HMOST-Algorithm

$G = (V, E)$: graph, C: cost-vector, s: population size, p: number of generations MST: efficient spanning trees, ND_{HMOST}: non-dominated points Encoding scheme setup;
Generate starting population P;
Initialize generation counter $iter := 0$, $Q := \emptyset$;
$iter \leq p$ Apply crossover-repair operator on P, store offsprings in Q;
Apply diversified mutation operator to elements in Q; $P := P \cup Q$;
Apply dynamic selection operator using crowding distance to get next population L;
$P := L$;
$iter := iter + 1$; Determine set $ST(F1)$ of trees in first front $F1$ of P;
$MST := ST(F1)$;
Extract $P(MST)$ from MST;
Apply Multi-VNS to obtain MST and ND_{HMOST};
MST and ND_{HMOST}

5. Computational experiments

The goal of this work is to show that the hybridization of the adapted NSGA-II metaheuristic with an exact method used locally, coupled with Multi-VNS Strategy, is able to give a better approximation of the Pareto front of the MOST problem than adapting a basic metaheuristic.

Parameter Settings: All experiments are conducted using *MATLAB R2018a* on a laptop with an *Intel Core i5* processor and *8 GB of RAM* and run on connected graphs. The HMOST-Algorithm uses the following parameters:

Population size:	$s = 100$
Number of generations:	$p = 50$
Crossover rate:	$\rho = 0.80$
Mutation rate:	$pm = 0.20$
Neighborhood structures in Multi-VNS:	$k = 1, 2, 3$
Stopping criterion (Multi-VNS):	20 iterations.

5.1. Comparison with the Pareto front

The first part of the experimental study is dedicated to the comparison between the Pareto front approximated by our proposed method and the Pareto front of the problem instance. Let ND_{HMOST} be the set of non-dominated points produced by the HMOST-Algorithm, and let $\alpha = \frac{|ND_{HMOST}|}{|ND|}$, where ND denotes the set of all non-dominated points. This ratio measures how well the HMOST-Algorithm approximates the Pareto front.

5.1.1. Instances description and computational results

1- Benchmark instances [20]: the results presented in Table 1 show that our approximate approach, HMOST-Algorithm, is able to recover on average 81.83% of the globally non-dominated solutions (i.e., those belonging to the Pareto front obtained by the exact method) in the three-criteria case and 80.16% in the four-criteria case. Approximately 20% of the remaining solutions are dominated.

Tests	n	m	$r = 3$			$r = 4$		
			$\|ND\|$	$\|ND_{HMOST}\|$	α	$\|ND\|$	$\|ND_{HMOST}\|$	α
*b*01	50	62	110	138	0.82	310	299	0.79
*b*05	50	100	712	946	0.80	923	1063	0.82
*b*07	75	94	340	377	0.83	402	410	0.80
*b*10	75	150	541	910	0.81	613	987	0.80
*b*13	100	125	101	120	0.84	304	412	0.81
*b*14	100	125	87	103	0.81	283	318	0.79

Table 1: *Comparison between Pareto front and HMOST-Algorithm for* $r = 3$ *and* $r = 4$.

2- SPACYC-Based Instance Generator [21]: The benchmark instances used in this study were kindly provided in the referenced article. We used 3 and 4 criteria with graphs ranging from 5 to 14 nodes. For each n, graphs with $m = n \cdot d$ edges, where $d \in \{5, 10, 15, 20\}$, were generated. The cost vectors were independently drawn from a uniform distribution over $[0, 100]$.

Table 2 shows that HMOST-Algorithm identified $81,7\%$ of non-dominated points for 3 criteria and $81,5\%$ for 4 criteria in average.

n	$r=3$			$r=4$		
	$\lvert ND\rvert$	$\lvert ND_{\text{HMOST}}\rvert$	α	$\lvert ND\rvert$	$\lvert ND_{\text{HMOST}}\rvert$	α
5	16.65	18.30	0.83	29.30	38.03	0.83
6	42.90	58.08	0.89	118.60	124.44	0.84
7	92.10	112.00	0.81	269.65	285.62	0.83
8	135.15	160.71	0.80	661.60	622.67	0.79
9	256.80	245.56	0.82	1370.95	1404.47	0.81
10	458.20	508.10	0.82	3629.15	3283.79	0.86
11	549.10	601.12	0.81	5426.50	6971.32	0.85
12	910.05	920.96	0.81	10985.80	12437.51	0.79
13	1206.20	1034.67	0.80	14881.60	15509.83	0.80
14	1909.00	2102.32	0.78	24679.75	22368.53	0.75

Table 2: *Comparison between exact Pareto front and HMOST-Algorithm for $r=3$ and $r=4$.*

3- Random instances: Graphs are generated using the Erdős–Rényi model [10], where a fixed number of vertices is defined and edges are added with a given probability p in which a fixed number of vertices is specified and edges are added with a given probability p. The cost vectors associated with the edges have dimensions 5 and 7, and their components are uniformly distributed within the interval $[-50, 100]$.

We can see that over 81% of non-dominated points have been found by the HMOST-Algorithm. We considered it useful to conduct an experiment on randomly generated instances in order to confirm the satisfactory results found previously, and indeed, the obtained results in Table 3 show that HMOST-Algorithm manages to generate nearly 80% of non-dominated points on average.

Tests	r	n	m	$\lvert ND\rvert$	$\lvert ND_{HMOST}\rvert$	α
Test1	5	20	[46,55]	74,80	307,20	0,81
Test2	5	30	[55,65]	930,30	980,50	0,79
Test3	5	50	[65,75]	1300,40	1612,06	0,77
Test4	5	100	[125,130]	627,39	1031,02	0,82
Test5	5	300	[325,340]	1267,40	1560,46	0,78
Test6	5	400	[425,440]	3427,94	6036,48	0,80
Test7	7	7	21	342,10	485,00	0,80
Test8	7	8	28	478,12	677,81	0,82

Table 3: *Comparison between Pareto front and HMOST-Algorithm using random instances with 5 and 7 criteria.*

5.2. Comparison with approximate methods

Comparative studies with randomly generated instances of cost-vectors of dimensions two, three and five, uniformly distributed in the interval $[-50, 100]$ are carried out for the three algorithms. The results are presented in Tables 5 and 6 respectively.
It should be noted that the number p of iterations is fixed as follows: 30 populations for the adapted NSGA-II procedure, 20 for the Multi-VNS strategy applied to each solution T in $P(MST)$, and 50 for each of the KEA and (NSGA II+PLS) algorithms.

Aspect	**HMOST**	**KEA**	**NSGA-II + PLS**
Chromosome encoding	edge-set	edge-set	Prüfer encoding
Extreme MSTs in the initial population	yes	yes	yes
Crossover operator	two-point crossover	not used	n-point crossover
Crossover probability	0.8	0	0.8
Mutation operator	k-opt, $k = 1, 2, 3$ solved by exact multi-objective method	knowledge-based mutation	controller-random mutation
Mutation probability	≤ 0.2	1	0.02
Hybridization strategy	NSGA-II + locally exact method + VNS	NSGA6II + k-best	NSGA-II + Pareto Local Search

Table 4: *Comparison of algorithmic innovations in HMOST, KEA, and NSGA-II+PLS.*

1 - Comparison using set coverage metric

This metric is used to compare two sets of potentially efficient solutions. The reference set, denoted REF, represents the set of non-dominated points obtained by combining the solutions from the two methods being compared.

The results are compared on average using the proportional measure to calculate the number of obtained non-dominated points of a given method belonging to the REF set [7].

We denote by $\alpha_i = (NDi \cap REF)/(REF)$ the proportion of solutions of the ND_i set belonging to REF set for $i \in \{HMOST, KEA, NSGAPLS\}$, where ND_{HMOST}, ND_{KEA} and $ND_{NSGAPLS}$ are the sets of non-dominated points obtained by HMOST, KEA-Algorithm and (NSGA-II+PLS)-Algorithm), respectively.

For each problem instance, we compare the non-dominated points produced by the two algorithms under evaluation.

K_n	r	$\lvert REF\rvert$	$\lvert ND_{HMOST}\rvert$	α_1	$\lvert ND_{KEA}\rvert$	α_2
$K80$	3	1700,02	1453,25	0,88	989,05	0,25
$K100$	3	4428,27	4324,80	0,73	3843,50	0,28
$K200$	3	9238,91	7277,00	0,87	7986,24	0,31
$K80$	5	3320,81	2878,10	0,86	2230,69	0,25
$K100$	5	8026,11	6695,08	0,83	4870,14	0,30
$K200$	5	10027,00	8783,60	0,87	6960,65	0,23

Table 5: *Comparison between HMOST-Algorithm and KEA-Algorithm.*

K_n	r	$\lvert REF\rvert$	$\lvert ND_{HMOST}\rvert$	α_1	$\lvert ND_{NSGAPLS}\rvert$	α_3
$K80$	2	1528,43	1213,9	0,85	1001,31	0,32
$K100$	2	5034,13	4421,76	0,82	3843,50	0,29
$K80$	3	2616,77	2027,08	0,79	1109,65	0,38
$K100$	3	9123,45	7392,92	0,84	5769,23	0,31

Table 6: *Comparison between HMOST-Algorithm and (NSGA-II + PLS)-Algorithm.*

Each algorithm is executed ten times for all instances of the problem. Therefore, we have found it useful to present in each row of Tables 5 and 6 the average results obtained across all runs.

Notice that for all instances, the HMOST-Algorithm outperforms the KEA-Algorithm by grabbing 84% and the (NSGA-II+PLS)-Algorithm by grabbing 83% from the set REF. Hence, it determines a greater number of solutions in the REF set compared to the KEA and (NSGA-II+PLS) Algorithms whose contributions are only 27% and 32% respectively.

2 - Comparison using two performance metrics

To evaluate the quality of the approximated points, we rely on two widely used indicators: the Hypervolume (HV) and the Inverted Generational Distance (IGD).

Hypervolume (HV) [23]: measures the volume of the objective space dominated by the approximated front with respect to a reference point Z_{ref} (commonly $Z_{\text{ref}} = \{0\}^k$ for k objectives). It reflects both convergence to the Pareto front and solution diversity. A higher HV indicates better performance.

Inverted Generational Distance (IGD) [7]: quantifies the average distance from each point in the Pareto front ND to its nearest solution in the approximated set A. Defined as:

$$\text{IGD}(ND, A) = \left(\frac{1}{|ND|} \sum_{s \in ND} d_s^2 \right)^{1/2}. \tag{2}$$

where d_s is the Euclidean distance between $s \in ND$ and its closest point in A. Lower IGD values indicate better convergence.

a - Comparison with the Pareto front

Instance	Hypervolume (HV)			IGD		
	KEA	NSGA-II+PLS	HMOST	KEA	NSGA-II+PLS	HMOST
Test4	45037	43572	67001	0.42	0.55	0.23
Test5	108786	127659	138716	0.87	0.76	0.38
Test6	442700	413215	603604	1.25	1.32	0.60

Table 7: *Comparison of algorithms on Hypervolume (HV) and IGD indicators.*

In order to assess the convergence of the obtained Pareto fronts of the three methods to be compared towards the Pareto front, we computed the HV and IGD metrics values. The obtained results are resumed in Table 7. It is obvious that the obtained values HV and IGD of each instance using HMOST-Algorithm are better than those obtained by the other two algorithms.

b - Comparison with reference set

Instance	r	Hypervolume (HV)			IGD		
		KEA	NSGA-II+PLS	HMOST	KEA	NSGA-II+PLS	HMOST
K50	3	9.71	9.66	9.87	0.74	1.07	0.39
K100	3	39.02	40.75	41.78	2.73	2.18	0.69
K200	3	82.97	83.32	88.99	9.50	10.54	3.53
K50	4	5869.6	5596.8	7326.2	2.30	6.41	1.82
K100	4	86807.1	82412.5	93812.2	7.30	9.84	3.79
K200	4	85955.4	82399.7	98077.1	4.75	6.12	2.77

Table 8: *Comparison of algorithms on HV and IGD indicators (values in 10^5 for HV).*

Table 8, for both 3-objective and 4-objective instances, HMOST-Algorithmachieves the best performance in the majority of the tested cases. These results demonstrate the strong ability of HMOST-Algorithm to generate high-quality Pareto fronts with both better convergence and distribution, especially as the problem size and number of objectives increase.

6. Conclusion

This study aimes to provide a comprehensive depiction of the Pareto front for the MOST problem. To address this, we introduced the HMOST-Algorithm, which operates in two distinct phases. Initially, the hybrid approach combines the NSGA-II algorithm that uses locally an exact mutation operator to obtain the first Pareto front. In the second phase, we apply the Multi-VNS Strategy to improve a subset of the previous obtained solutions . By leveraging a specialized mutation operator together with the the Multi-VNS Strategy, the HMOST-Algorithm becomes a flexible optimization framework capable to controlling both phases through predefined probabilities.

A comparative study demonstrated that, on average, the proposed HMOST-Algorithm identifies nearly 81% of the non-dominated points. This confirms that the generated front closely approximates the Pareto front. In addition to overcoming the inability of the exact method to solve large instances, given the explosive complexity of the MOST problem, the proposed algorithm also demonstrates excellent performance compared to the KEA and (NSGA-II+PLS) algorithms in comparative studies on complete graphs with up to 200 nodes and cost-vectors with dimensionalities between 2 and 7.

Considering the experimental results, the HMOST-Algorithm consistently achieves lower IGD values, indicating stronger convergence toward the Pareto front. Additionally, it obtains higher HV values, reflecting better diversity and distribution of solutions across the objective space. This indicates that the Pareto front approximated by the HMOST-Algorithm is closer to the Pareto front and exhibits broader coverage compared to those obtained by the two cited algorithms, which, to the best of our knowledge, are the most recent available methods. However, while the algorithm performs well on graphs of up to 200 nodes, the computational cost of the exact k-opt mutation and Multi-VNS steps could become a bottleneck for very large-scale graphs (e.g., 1000+ nodes). Moreover, although KEA and NSGA-II+PLS are relevant benchmarks, additional recent multi-objective metaheuristics (e.g., NTGA2, theta-DEA, MOEAD) were not included in this comparative analysis. To further enhance the HMOST-Algorithm's scalability, it is advisable to explore parallelization of the exact mutation and Multi-VNS procedures. Additionally, investigating approximate k-opt variants or incorporating surrogate modeling techniques can help reduce the computational burden associated with large neighborhood structures.

References

[1] Amorosi, L. and Puerto, J. (2022). Two-phase strategies for the bi-objective minimum spanning tree problem. Intl. Trans. in Op. Res, 29(6), 3435–3463. doi: 10.1111/itor.13120.

[2] Arroyo, J. E. C., Vieira, P.S. and Vianna, D.S. (2008). A GRASP algorithm for the multi-criteria minimum spanning tree problem. Ann Oper Res, 156, 125–133. doi: 10.1007/s10479-007-0263-4.

[3] Boumesbah, A. and Chergui, M. E. - A. (2016). An exact method to generate all nondominated spanning trees. RAIRO Operations Research, 50(4-5), 857–867. doi: 10.1051/ro/2016060.

[4] Burmeister, S. C. (2025). A memetic NSGA-III for green flexible production with real-time energy costs and emissions. Croatian Operational Research Review, 16(2), 99–111. doi: 10.17535/crorr.2025.0009.

[5] Camerini, P. M., Galbiati, G. and Maffioli, F. (1984). The complexity of weighted multi-constrained spanning tree problems. In Lovász, L. (Ed.) Colloquium on the Theory of Algorithms (53-101), North-Holland.

[6] Correia, P., Paquete, L., Burton, B. A. and Figueira, J. R. (2021). Finding multi-objective supported efficient spanning trees. Computational Optimization and Applications, 78, 491–528. doi: 10.1007/s10589-020-00251-6.

[7] Van Veldhuizen, D. A. and Lamont, G. B. (2000). On measuring multi-objective evolutionary algorithm performance. Evolutionary Computation, 1, 204–211. doi: 10.1109/CEC.2000.870296.

[8] David-Moradkhan, M. and Browne, W. (2010). Evolutionary Algorithms for the Multi-Criterion Minimum Spanning Tree Problem. In Tenne, Y. and Goh, C.-K. (Eds.) Computational Intel. in Expensive Opti. Prob., ALO, 423–452.

[9] Deb, K., Agrawal, S., Pratap, A. and Meyarivan, T. (2002). A fast and elitist multi-objective genetic algorithm for multi-objective optimization. IEEE Trans. Evol. Comput, 6(2), 181–197. doi: 10.1109/4235.996017.

[10] Erdös, P. and Rényi, A. (1960). On the evolution of random graphs. Publ. Math. Inst. Hung. Acad. Sci., 5, 17–61.

[11] Fernandes, F. C., Goldbarg, E. F. G., Maia, S. M. D. M. and Goldbarg, M. C. (2020). Empirical study of exact algorithms for the multi-objective spanning tree. Comput. Optim. Appl, 75(2), 561–605. doi: 10.1007/s10589-019-00154-1.

[12] Gottlieb, J., Julstrom, B. A., Rothlauf, F. and Raidl, G. R. (2001). Prufer numbers: A poor representation of spanning trees for evolutionary search. In: Proceedings of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, San Francisco, 3, 343–350.

[13] Guo, W., Chen, G., Feng, X. and Yu, L. (2007). Solving multi-criteria minimum spanning tree problem with discrete particle swarm optimization. In: Proc. 3rd Int. Conf. Nat. Comput. (ICNC 2007), 3, 471–478. doi: 10.1109/ICNC.2007.673.

[14] Han, L. and Wang, Y. (2005). A novel genetic algorithm for multi-criteria minimum spanning tree problem. In Hao, Y. et al. (Eds.), CIS 2005, LNCS, 3801, 297–302. doi: $10.1007/11596448_46$.

[15] Hansen, P. and Mladenović, N. (1997). Variable neighborhood search. Computers and Operations Research, 24(11), 1097–1100. doi: 10.1016/S0305-0548(97)00031-2.

[16] Knowles, J. and Corne, D. (2002). Enumeration of Pareto optimal multi-criteria spanning trees - a proof of the incorrectness of Zhou and Gen's proposed algorithm. European Journal of Operational Research, 143(3), 543–547. doi: 10.1016/S0377-2217(01)00346-0.

[17] Majumder, S., Barma, P. S., Biswas, A., Banerjee, P., Mandal, B. K., Kar, S. and Ziemba, P. (2022). On Multi-Objective Minimum Spanning Tree Problem under Uncertain Paradigm. Symmetry, 14(1), 106. doi: 10.3390/sym14010106.

[18] Maristany de las Casas, P., Sedeño-Noda, A. and Borndörfer, R. (2023). New Dynamic Programming Algorithm for the multi-objective Minimum Spanning Tree Problem. Computers and Operations Research, 173, 106852. doi: 10.1016/j.cor.2024.106852.

[19] Párraga-Álava, P., Dorn, M. and Inostroza-Ponta, M. (2017). Using Local Search Strategies to Improve the Performance of NSGA-II for the Multi-Criteria Minimum Spanning Tree Problem. IEEE Congress on Evolutionary Computation, 1119-1126. doi: 10.1109/CEC.2017.7969432.

[20] Pugliese, L., Guerriero, F. and Santos J. L. (2015). Dynamic programming for spanning tree problems: application to the multi-objective case. Optim. Lett, 9(3), 437–450. doi: 10.1007/s11590-014-0759-1.

[21] Santos, J. L., Pugliese, L. D. and Guerriero, F. (2018). A new approach for the multi-objective minimum spanning tree. Computers and Operations Research, 98, 69–83. doi: 10.1016/j.cor.2018.05.007.

[22] Zhou, G. and Gen, M. (1999). Genetic algorithm approach on multicriteria minimum spanning tree problem. European Journal of Operational Research, 114(1), 141–152. doi: 10.1016/S0377-2217(98)00016-2.

[23] Zitzler, E. and Thiele, L. (1999). Multi-objective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. IEEE Transactions on Evolutionary Computation, 3(4), 257-271. doi: 10.1109/4235.797969.