Runtime Analysis of Evolutionary Algorithms for the Depth Restricted (1,2)-Minimum Spanning Tree Problem

Feng Shi
Central South University
School of Computer Science and Engineering
Changsha, China

Frank Neumann
The University of Adelaide
School of Computer Science
Adelaide, Australia

Jianxin Wang
Central South University
School of Computer Science and Engineering
Changsha, China

ABSTRACT

The Minimum Spanning Tree problem is a well-known combinatorial optimization problem, which has attracted much attention from the researchers in the field of evolutionary computing. Within the paper, a constrained version of the problem named Depth Restricted (1-2)-Minimum Spanning Tree problem is considered in the context of evolutionary algorithms, which had been shown to be NP-hard. We separately investigate the expected time (i.e., the expected number of fitness evaluations) of the (1+1) EA, the Multi-Objective Evolutionary Algorithm and its two variants adapted to the constrained version, to obtain an approximate solution with ratio 2 or $\frac{3}{2}$ with respect to several different fitness functions. In addition, we observe a close connection between the constrained version and the Set Cover problem, and present a simple evolutionary algorithm for the 3-Set Cover problem. Based on the approximate solution returned by our evolutionary algorithm for the 3-Set Cover problem, an approximate solution with ratio better than $\frac{3}{2}$ for the constrained version can be constructed.

CCS CONCEPTS

• **Mathematics of computing** → *Evolutionary algorithms*; • **Theory of computation** → *Random search heuristics.*

KEYWORDS

minimum spanning tree, evolutionary algorithm, time analysis, depth restricted (1,2)-minimum spanning tree

ACM Reference Format:
Feng Shi, Frank Neumann, and Jianxin Wang. 2019. Runtime Analysis of Evolutionary Algorithms for the Depth Restricted (1,2)-Minimum Spanning Tree Problem. In *Foundations of Genetic Algorithms XV (FOGA '19), August 27–29, 2019, Potsdam, Germany.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3299904.3340314

FOGA '19, August 27–29, 2019, Potsdam, Germany

ACM ISBN 978-1-4503-6254-2/19/08...$15.00
https://doi.org/10.1145/3299904.3340314

1 INTRODUCTION

Over the past decades, evolutionary algorithms have been extensively studied to solve the combinatorial optimization problems abstracted from real applications in various areas, including engineering and economics. Lots of progress has been achieved in the theoretical analysis of the behavior of evolutionary algorithms, in particular, for the well-known Traveling Salesperson problem [20, 35], Vertex Cover problem [12, 18, 28, 29, 32], Knapsack problem [22, 31, 36], Makespan Scheduling problem [26, 34], and Minimum Spanning Tree problem [8, 24, 25], et al.

Within the paper, we study a constrained version of the Minimum Spanning Tree problem (abbr. MSTP). Thus in the following, we first introduce the background of the problem and related work in the field of evolutionary computing. Given an edge-weighted graph G, MSTP asks for a connected subgraph of G that contains all vertices in G, without any cycle and with the minimum cost, where the cost is the sum of the weights on the edges in the subgraph. It is well-known that the problem is polynomial solvable, using the classic Prim's algorithm [16] or Kruskal's algorithm [19].

Neumann and Wegener [24] studied the performance of the Randomized Local Search (abbr. RLS) and (1+1) EA for MSTP. Using the fitness function that penalizes the disconnectivity of the corresponding subgraph induced by the search point, the expected time of the two algorithms to obtain an optimal solution were shown to be bounded by $O(m^2(\log n + \log w_{max}))$, where m and n denote the numbers of edges and vertices in the considered graph, and w_{max} denotes the maximum weight that the edges have. Later Neumann and Wegener [23] studied the performances of algorithm SEMO and GSEMO with respect to a two-objective fitness function, which consists of the number of connected components in the corresponding subgraph induced by the search point and the cost of the chosen edges. They showed that the expected time of the two algorithms can be bounded by $O(mn(n + \log w_{max}))$.

Kratsch et al. [17] investigated the NP-hard problem Maximum Leaf Spanning Tree by evolutionary algorithms in the context of fixed parameter tractability [9, 11], where the maximum number of leaves is considered as the parameter. Corus et al. [8] examined the NP-hard *Generalized* MSTP and analyzed two approaches in the context of fixed parameter tractability and bi-level optimization. They showed that their specific (1+1) EA working with the spanning nodes representation is not a fixed-parameter evolutionary algorithm, whereas the one working with the global structure representation is. Neumann [21] considered the multi-objective version of MSTP, where each edge e in the input graph has a weight vector $w(e) = (w_1(e), \ldots, w_k(e))$, and $w_i(e)$ is a positive integer for all $1 \leq i \leq k$. The problem asks for a Pareto set that contains a minimum spanning tree with respect to each w_i. They showed that a simple evolutionary algorithm can obtain a population that is a 2-approximation of the Pareto front. Besides the work mentioned above, there is an active research line on evolutionary algorithms for the Bounded Diameter MSTP [13, 14, 30].

The constrained version of MSTP considered in this paper was abstracted from the realistic scenarios of telecommunication network construction. Consider the network construction for a lot of cities for example. A city is always designated as the center of the network (such as the capital of a nation or province), and the key challenge is the selection of some cities to be intermediate transmitters to connect the central city with all the other cities, so that the cost of the connections is minimized. Alfandari and Paschos [1] modeled the realistic problem as follows: given a complete graph $G = (\{r\} \cup V, E, W)$, where each vertex in $\{r\} \cup V$ corresponds to a city (the specific vertex r corresponds to the designated central city), and the weight function W is defined on the edge-set E; the aim is to find a subset $E' \subset E$ with the minimum cost such that the subgraph $G[E']$ obtained by removing all edges in $E \setminus E'$ from G is connected, and for any vertex $v \in V$, the shortest path connecting v and r in $G[E']$ contains at most two edges. Since the cost of the connections should be minimized, the subgraph $G[E']$ is actually a spanning tree. Thus the problem is named *Depth Restricted Minimum Spanning Tree problem* (abbr. DR-MSTP) in the paper. In addition, Alfandari and Paschos [1] showed that the problem is NP-hard and cannot be approximated with a ratio better than $O(\ln n)$ ($n = |V|$, i.e., there are $n + 1$ vertices in the graph including the specific vertex r).

The famous Traveling Salesperson problem and Steiner Tree problem have been studied extensively under the assumption that the weight on each edge in the considered graph is either 1 or 2 [2–5, 27] (one can consider that the weights on the edges are not well-defined, just "large" and "small"). Thus Alfandari and Paschos [1] also investigated DR-MSTP under this assumption (the NP-hardness also holds under the assumption), and gave an approximation algorithm with ratio 1.25. In the remainder of the paper, we consider the problem under the same assumption, which is called DR-(1,2)-MSTP, in the context of evolutionary algorithms. More specifically, we study the expected time (i.e., the expected number of fitness evaluations) of the evolutionary algorithms considered in the paper to obtain an approximate solution, and the corresponding approximate ratios.

Firstly, we compare the performance of the (1+1) EA to get a 2-approximate solution for DR-(1,2)-MSTP, with respect to eight different fitness functions. Due to the elitist selection mechanism of the (1+1) EA, if the algorithm maintains a solution corresponding to a spanning tree violating the depth restriction, then it needs to "swap" two edges in the spanning tree (i.e., takes expected time $O(m^2)$ if considering the edge-representation, where $m = |E|$) to improve the solution without introducing any cycle and causing the disconnectivity, with respect to some fitness function settings. Thus we also study the Multi-Objective Evolutionary Algorithm (abbr. MOEA) adapted to DR-(1,2)-MSTP, aiming to avoid the "swap" operation by maintaining a population. The population of the MOEA keeps a solution with i edges for each $i \in [0, n]$, i.e., the population size is at most $n + 1$. Apparently, the large population of the MOEA may slow down its optimization process, thus we consider two variants of the MOEA, named MOEA-1 and MOEA-2, where each

Ratio		2	$\frac{3}{2}$
(1+1) EA	f_1	$O(m^2 \log n)$	$O(m^6 n)$
	f_2	$O(m^2 \log n)$	$O(m^6 n)$
	f_3	$O(m^2 \log n)$	$O(m^6 n)$
	f_1'	$O(m^2 \log n)$	$O(m^6 n)$
	f_2'		
	f_3'		
	f_4	$O(m \log n)$	$O(m^6 n)$
	f_5	$O(m \log n)$	$O(m^6 n)$
MOEA		$O(mn \log n)$	$O(m^6 n^2)$
MOEA-1		$O(mn)$	$O(m^6 n)$
MOEA-2		$O(m \log n)$	$O(m^4 n)$

Table 1: Overview of results, where $m = \Theta(n^2)$. Upper bounds on the expected time of the (1+1) EA, the Multi-Objective Evolutionary Algorithm (MOEA) and its two variants (MOEA-1 and MOEA-2) to get a solution of DR-(1,2)-MSTP with approximate ratio 2 or $\frac{3}{2}$. In particular, for the (1+1) EA with fitness function f_2' or f_3', we show that once they accept a specific solution, then they need exponential expected time to get an improved solution.

of them maintains a population with at most two solutions. We show that the MOEA-1 and MOEA-2 can efficiently emulate the "swap" operation and the local search operation flipping more than two edges at the same time, respectively. Afterwards, using the local search strategy, we analyze the expected time of the (1+1) EA, the MOEA and its two variants to get an improved $\frac{3}{2}$-approximate solution. A summary of the obtained results is given in Table 1.

Finally, we reformulate DR-(1,2)-MSTP in form of the classical Set Cover problem following the work of Alfandari and Paschos [1], and give evolutionary computing another opportunity to solve the problem, leading to a better approximate ratio.

The rest of the paper is organized as follows. Section 2 introduces related definitions, and Section 3 presents the four considered algorithms (1+1) EA, the MOEA and its two variants. The detailed analysis on the performance of the four algorithms to obtain an approximate solution with ratio 2 and $\frac{3}{2}$ is given in Section 4 and Section 5, respectively. In Section 6 we reduce DR-(1,2)-MSTP to the Set Cover problem, and give an improved analysis. We use Section 7 to conclude this work.

2 PRELIMINARIES

A graph is *complete* if there exists an edge between any two vertices in the graph. Consider a complete edge-weighted graph $G = (\{r\} \cup V, E, W)$, where r is a specific vertex (we simply call it the *root* of G in the remaining context), $V = \{v_1, \ldots, v_n\}$, $E = \{e_1, \ldots, e_m\}$, and $W : E \longrightarrow \mathbb{N}$ (note again that G has $n + 1$ vertices). For a vertex v in G, denote by $N_i(v)$ ($i \in \mathbb{N}$) the set containing all the vertices v' in G with $W([v, v']) = i$. For an edge-subset $E' \subset E$, denote by $G[E']$ the graph obtained by removing all edges in $E \setminus E'$ from G. That is, $G[E']$ and G have the same vertex-set $\{r\} \cup V$.

A *spanning tree* of G is a subgraph of G that connects all vertices in G, and has no cycle. A *minimum spanning tree* of G is a spanning tree with the minimum weight, where the weight of a spanning tree is defined as the sum of weights on its edges. In other words, the *Minimum Spanning Tree problem* (abbr. MSTP) on G looks for an edge-subset E^* of E such that all vertices of $\{r\} \cup V$ are connected in $G[E^*]$, and $\sum_{e \in E^*} W(e)$ is minimized.

Thus the search space on which we study the behavior of evolutionary algorithms for MSTP in the paper, consists of all bit-strings with fixed length m. For any solution $x = x_1 \ldots x_m$, edge e_i ($1 \le i \le m$) is chosen iff $x_i = 1$. Denote by $|x|_1$ the number of 1-bits in x, i.e., the Hamming weight of x. Observe that any solution specifies a unique edge-subset of E, which is denoted by $E(x)$ (the cardinality of $E(x)$ equals $|x|_1$). Hence for simplicity of notation, we let $G(x)$ be the same graph as $G[E(x)]$. Denote by

$$Cost(x) = \sum_{i=1}^{m} W(e_i) \cdot x_i$$

the sum of weights on these chosen edges, by $C_r(x)$ the connected component in $G(x)$ that contains the root r, by $N_{\mathrm{cc}}(x)$ the number of connected components in $G(x)$, and by $N_{\mathrm{cc}}^{\ge 1}(x)$ the number of connected components in $G(x)$ that have more than one vertex, except the one $C_r(x)$.

Given a connected component C in $G(x)$ that does not contain r, if there exists a vertex v in C such that all the other vertices in C are the neighbors of v, and there is no edge between the neighbors of v, then the connected component is a *claw component*; otherwise, it is a *non-claw component*. That is, both the *singleton* component (consists of a vertex) and *singe-edge* component (consists of an edge with two vertices) are claw components. Note that if a claw component has more than 2 vertices, then the vertex with degree greater than 1 is the *center* of the claw component; if it has only 2 vertices v and v', then the vertex v with $W([v,r]) \le W([v',r])$ is the *center* of the claw component; if it has only 1 vertex, then itself is the *center*. Denote by $N_{\mathrm{cc}}^{\mathrm{c}}(x)$ and $N_{\mathrm{cc}}^{\mathrm{nc}}(x)$ the numbers of claw components and non-claw components in $G(x)$, respectively.

Given two vertices v_1 and v_2 in $G(x)$, the *distance* between them, denoted by $d_{G(x)}(v_1, v_2)$, is the number of edges in the shortest path connecting v_1 and v_2 in $G(x)$ if they are in the same connected component of $G(x)$; otherwise, $+\infty$. Denote by $N_{d \ge i}(x)$ (resp., $N_{d > i}(x)$) the number of vertices $v \in V$ with $d_{G(x)}(v, r) \ge i$ (resp., $d_{G(x)}(v, r) > i$), where $i \in \mathbb{N}$. Thus $N_{d > n}(x)$ denotes the number of vertices that are not in the same connected component $C_r(x)$ with r in $G(x)$ (as G contains $n+1$ vertices). More specifically, we let $N_{n \ge d > 2}(x) = N_{d>2}(x) - N_{d>n}(x)$ and $N_{2 \ge d > 0}(x) = N_{d>0}(x) - N_{d>2}(x)$.

In the paper we consider a constrained variant of MSTP, named *Depth Restricted (1-2)-Minimum Spanning Tree problem* (abbr. DR-(1-2)-MSTP). The problem is considered on an edge-weighted complete graph $G = (\{r\} \cup V, E, W)$ with $W : E \to \{1, 2\}$, looking for a minimum spanning tree T of G such that $d_T(v, r) \le 2$ for any vertex $v \in V$. As the vertex r can be regarded as the root of T, thus the *depth* of the tree T is upper bounded by 2.

A solution x of DR-(1-2)-MSTP on G is *feasible* if $G(x)$ is a spanning tree, and $d_{G(x)}(v, r) \le 2$ for any $v \in V$; otherwise, *infeasible*.

3 ALGORITHMS

We present four evolutionary algorithms in the paper, namely, the (1+1) EA, the Multi-Objective Evolutionary Algorithm (abbr. MOEA) and its two variants (MOEA-1 and MOEA-2), and study their expected time (i.e., the expected number of fitness evaluations they need) to achieve an approximate solution of DR-(1-2)-MSTP on the complete edge-weighted graph G.

3.1 (1+1) EA

Algorithm 1: (1+1) EA

```
initialize solution x = 0^m;
while stopping criterion not met do
    y ← flip each bit of x independently w/ probability 1/m;
    if f(y) ≤ f(x) then
        x ← y;
```

The (1+1) EA (given in Algorithm 1) starts with the solution 0^m. In each iteration, the algorithm generates an offspring using the standard mutation operator on the maintained solution, then chooses the one with better fitness from them to maintain. To obtain a feasible solution of DR-(1-2)-MSTP on G, the fitness function $f(x)$ that decides the fitness of solution x should penalize the factors that may cause the infeasibility of x, namely, the disconnectivity of the graph $G(x)$ (the term $N_{\mathrm{cc}}(x) - 1$), the presence of vertices whose distances to r are greater than 2 (the term $N_{d>2}(x)$) and cycles (the term $N_{\mathrm{cc}}(x) + |x|_1 - n - 1$). However, the setting of the scales of the penalties on these factors has a direct impact on the performance of the algorithm. Thus we consider three fitness functions as follows, with different settings.

$$\begin{aligned}
f_1(x) &= Cost(x) + 4m^2 \cdot (N_{\mathrm{cc}}(x) - 1) + 2m \cdot N_{d>2}(x) \\
&+ 4m^2 \cdot (N_{\mathrm{cc}}(x) + |x|_1 - n - 1) \\
f_2(x) &= Cost(x) + 2m \cdot (N_{\mathrm{cc}}(x) - 1) + 4m^2 \cdot N_{d>2}(x) \\
&+ 4m^2 \cdot (N_{\mathrm{cc}}(x) + |x|_1 - n - 1) \\
f_3(x) &= Cost(x) + 2m \cdot [(N_{\mathrm{cc}}(x) - 1) + N_{d>2}(x)] \\
&+ 4m^2 \cdot (N_{\mathrm{cc}}(x) + |x|_1 - n - 1)
\end{aligned}$$

For the fitness function $f_1(x)$, the scale of penalty on the disconnectivity of $G(x)$, is larger than that on the presence of vertices whose distances to r are greater than 2. Conversely, for the fitness function $f_2(x)$, the scale of penalty on the presence of vertices whose distances to r are greater than 2, is larger than that on the disconnectivity of $G(x)$. The fitness function $f_3(x)$ considers the case that the scale of penalty on the presence of vertices whose distances to r are greater than 2 is the same as that on the disconnectivity of $G(x)$. Note that all the three functions f_1, f_2 and f_3 have the same scale of penalty on the presence of cycles, moreover, the scale is always not less than that on the presence of vertices whose distances to r are greater than 2. That is because destroying a cycle by removing an edge may cause the presence of vertices whose distances to r are greater than 2, and the resulting solution cannot be accepted if the scale of penalty on the presence of cycles is less than that on the presence of vertices with distance to r greater than 2.

For a vertex $v \in V$ that is not in the same connected component $C_r(x)$ with r in $G(x)$, since it may be connected with other vertices in $V \setminus \{v\}$ such that the number of connected components in $G(x)$ is decreased, it is unnecessary to penalize the vertex v again due to the term $N_{d>2}(x)$. Thus we replace the term $N_{d>2}(x)$ in f_1, f_2, and f_3 with $N_{n \geq d>2}(x)$, and give the following three corresponding fitness functions.

$$\begin{aligned} f_1'(x) &= Cost(x) + 4m^2 \cdot (N_{\rm cc}(x) - 1) + 2m \cdot N_{n\geq d>2}(x) \\ &+ 4m^2 \cdot (N_{\rm cc}(x) + |x|_1 - n - 1) \\ f_2'(x) &= Cost(x) + 2m \cdot (N_{\rm cc}(x) - 1) + 4m^2 \cdot N_{n\geq d>2}(x) \\ &+ 4m^2 \cdot (N_{\rm cc}(x) + |x|_1 - n - 1) \\ f_3'(x) &= Cost(x) + 2m \cdot [(N_{\rm cc}(x) - 1) + N_{n\geq d>2}(x)] \\ &+ 4m^2 \cdot (N_{\rm cc}(x) + |x|_1 - n - 1) \end{aligned}$$

For the fitness function f_2', the penalty on the term $N_{\rm cc}(x) + |x|_1 - n - 1$ avoids the presence of cycles in $G(x)$, and that on the term $N_{n\geq d>2}(x)$ upper bounds the depth of the connected component $C_r(x)$ in $G(x)$. However, the structures of the connected components in $G(x)$ except $C_r(x)$ cannot be restricted. If there is a connected component C in $G(x)$ except $C_r(x)$ that is a non-claw component, then any approach that connects C and $C_r(x)$ would result in a new solution x' with $N_{n\geq d>2}(x') > N_{n\geq d>2}(x)$. That is, all connected components in $G(x)$ expect $C_r(x)$ should be claw components. Thus we add the term $N_{\rm cc}^{\rm nc}(x)$ into fitness function f_2', and give the following fitness function.

$$\begin{aligned} f_4(x) &= Cost(x) + 2m \cdot (N_{\rm cc}(x) - 1) \\ &+ 4m^2 \cdot [N_{n\geq d>2}(x) + N_{\rm cc}^{\rm nc}(x) + (N_{\rm cc}(x) + |x|_1 - n - 1)] \end{aligned}$$

Inspired by Kruskal's algorithm [19], we add the term $N_{\rm cc}^{>1}(x)$ into fitness function f_2', and utilize the penalty on it to restrict the locations of the new added edges. Recall that $N_{\rm cc}^{>1}(x)$ counts the number of connected components with more than one vertex in $G(x)$, except the one $C_r(x)$. Thus the new edges can only be added into the connected component $C_r(x)$. The corresponding fitness function is given as follows.

$$\begin{aligned} f_5(x) &= Cost(x) + 2m \cdot (N_{\rm cc}(x) - 1) \\ &+ 4m^2 \cdot [N_{n\geq d>2}(x) + N_{\rm cc}^{>1}(x) + (N_{\rm cc}(x) + |x|_1 - n - 1)] \end{aligned}$$

It is easy to see that once a feasible solution is found, then the elitist selection of the (1+1) EA with one of the above eight fitness functions bars it from adopting an infeasible solution ever again.

3.2 Multi-Objective Evolutionary Algorithms

The Multi-Objective Evolutionary Algorithm (abbr. MOEA, given in Algorithm 2) uses a vector-valued fitness function

$$f_{\rm M}(x) = [|x|_1, f_{\rm M}^2(x)],$$

where

$$f_{\rm M}^2(x) = Cost(x) + 2m \cdot [(N_{\rm cc}(x) - 1) + N_{d>2}(x)]\,.$$

As the first and second objectives of $f_{\rm M}(x)$ consider the Hamming weight of x and the number of connected components in $G(x)$, respectively, it is unnecessary to penalize the presence of cycles in $G(x)$ again. Thus the term $N_{\rm cc}(x) + |x|_1 - n - 1$ disappears from the second objective $f_{\rm M}^2(x)$ of $f_{\rm M}(x)$.

Given two solutions y and z, y *dominates* z with respect to $f_{\rm M}$ if $|y|_1 = |z|_1$ and $f_{\rm M}^2(y) \leq f_{\rm M}^2(z)$, written $y \succcurlyeq_{\rm MOEA} z$; y *strongly dominates* z if y dominates z but $y \neq z$, written $y \succ_{\rm MOEA} z$. Thus two solutions are comparable with respect to $f_{\rm M}$ only if they have the same Hamming weight.

The population S of the MOEA is initialized with $\{0^m\}$. In each iteration, the MOEA picks an individual x randomly from S, and generates an offspring y based on x using the standard mutation operator. If y has a Hamming weight between 0 and n, and is not strongly dominated by another solution in S, then all the solutions dominated by y in S are discarded, and y is included into S. Thus the size of S is upper bounded by $n + 1$, in which any two individuals are incomparable.

Algorithm 2: MOEA

```
S ← {0^m};
while stopping criterion not met do
    choose x ∈ S uniformly at random;
    y ← flip each bit of x independently w/ probability 1/m;
    if (0 ≤ |y|_1 ≤ n) ∧ (∄w ∈ S: w ≻_MOEA y) then
        S ← S \ {z ∈ S | y ≽_MOEA z};
        S ← S ∪ {y};
```

As the optimization process of the MOEA may be slowed down due to its large population, we consider its first variant, named MOEA-1 (given in Algorithm 3), whose population size can be bounded by a constant 2. The MOEA-1 uses the same fitness function $f_{\rm M}$ as MOEA, but differs at the notion of dominance between solutions, written $\succcurlyeq_{\rm MOEA-1}$, whose definition is inspired by that of the dominance $\succcurlyeq_{\rm MOEA-S}$ given in [33]. Given two solutions y and z, where $|y|_1$ and $|z|_1$ are required to be in $[0, n+1]$, if *at most one* of the values $|y|_1$ and $|z|_1$ is in $[n, n+1]$, then they are ordered lexicographically,

$$y \succcurlyeq_{\rm MOEA-1} z \iff (|y|_1 > |z|_1) \vee \Big(|y|_1 = |z|_1 \wedge f_{\rm M}^2(y) \leq f_{\rm M}^2(z)\Big)\,. \quad (1)$$

If *both* $|y|_1$ and $|z|_1$ are in $[n, n+1]$, then we set

$$y \succcurlyeq_{\rm MOEA-1} z \iff |y|_1 = |z|_1 \wedge f_{\rm M}^2(y) \leq f_{\rm M}^2(z). \quad (2)$$

Solution y *strongly dominates* z if $y \succcurlyeq_{\rm MOEA-1} z$ and $f_{\rm M}^2(y) < f_{\rm M}^2(z)$, written $y \succ_{\rm MOEA-1} z$.

Consequently, two bit strings y and z are incomparable if and only if both $|y|_1$ and $|z|_1$ are in $[n, n+1]$ and $|y|_1 \neq |z|_1$, implying that the size of the population maintained by the MOEA-1 can be bounded by 2. The purpose to keep such a population that can maintain two solutions with Hamming weights n and $n+1$ respectively at the same time, is that the solution with Hamming weight $n+1$ can be an intermediate to accelerate the optimization process of the solution with Hamming weight n.

Given a solution x, denote by $E_d(x)$ the subset of $E(x)$ with the minimum size such that replacing each edge $[v, p] \in E_d(x)$ with $[v, r]$ results in a solution x' with $|x|_1 = |x'|_1$ and $N_{n\geq d>2}(x') = 0$, where p is the neighbor of v that is in a shortest path connecting

Algorithm 3: MOEA-1

```
S ← {0^m};
while stopping criterion not met do
    choose x ∈ S uniformly at random;
    y ← flip each bit of x independently w/ probability 1/m;
    if (0 ≤ |y|_1 ≤ n + 1) ∧ (∄w ∈ S: w ≻_MOEA-1 y) then
        S ← S \ {z ∈ S | y ⪰_MOEA-1 z};
        S ← S ∪ {y};
```

v and r in $G(x)$. If $|x|_1 = n$, $N_{cc}(x) = 1$, and $|E_d(x)| = 1$, then we have the observation that a feasible solution based on x can be easily obtained by swapping the unique edge $[v, p]$ in $E_d(x)$ with the edge $[v, r]$. Thus we can use such a solution as intermediate to promote the optimization process of the maintained solutions (similar to the idea of the MOEA-1), and give the second variant of the MOEA, named MOEA-2 (given in Algorithm 4), using the following vector-valued fitness function.

$$f_{M2}(x) = [|E_d(x)|, f^2_{M2}(x)],$$

where

$$\begin{aligned} f^2_{M2}(x) &= Cost(x) + 2m \cdot (N_{cc}(x) - 1) \\ &+ 4m^2 \cdot [N^{\geq 1}_{cc}(x) + (N_{cc}(x) + |x|_1 - n - 1)] \end{aligned}$$

Since the first objective $|E_d(x)|$ of $f_{M2}(x)$ only considers the edges in the same connected component $C_r(x)$ with r, the second objective $f^2_{M2}(x)$ has a penalty on the term $N^{\geq 1}_{cc}(x)$ to let all edges in $G(x)$ be in the connected component $C_r(x)$.

Similar to the MOEA, given two solutions y and z, where $|E_d(y)|$ and $|E_d(z)|$ are required to be 0 or 1, y *dominates* z with respect to f_{M2} if $|E_d(y)| = |E_d(z)|$ and $f^2_{M2}(y) \leq f^2_{M2}(z)$, written $y \succeq_{\text{MOEA-2}} z$. Solution y *strongly dominates* z if y dominates z but $y \neq z$, written $y \succ_{\text{MOEA-2}} z$. Thus y and z are incomparable with respect to f_{M2} only if $|E_d(y)| + |E_d(z)| = 1$.

Algorithm 4: MOEA-2

```
S ← {0^m};
while stopping criterion not met do
    choose x ∈ S uniformly at random;
    y ← flip each bit of x independently w/ probability 1/m;
    if (0 ≤ |E_d(x)| ≤ 1) ∧ (∄w ∈ S: w ≻_MOEA-2 y) then
        S ← S \ {z ∈ S | y ⪰_MOEA-2 z};
        S ← S ∪ {y};
```

4 ANALYSIS OF DR-(1,2)-MSTP

As G contains $n + 1$ vertices and each edge in G has weight 1 or 2, the weight of a spanning tree in G ranges from n to $2n$. We have the observation given below.

Observation. Any feasible solution of DR-(1,2)-MSTP on G has an approximate ratio 2.

Given a solution x with $N_{cc}(x) + |x|_1 > n + 1$, the following lemma indicates that $G(x)$ contains $N_{cc}(x) + |x|_1 - n - 1$ edges that can be removed safely.

LEMMA 4.1. *Given a solution x with $N_{cc}(x) + |x|_1 > n + 1$, it contains $N_{cc}(x) + |x|_1 - n - 1$ 1-bits, each of whose flip results in a solution x' with $Cost(x') < Cost(x)$, $N_{cc}(x') = N_{cc}(x)$, and $N_{d>2}(x') = N_{d>2}(x)$.*

PROOF. Observe that $G(x)$ contains at least $N_{cc}(x) + |x|_1 - n - 1$ different cycles. Let C be an arbitrary cycle in $G(x)$ (note that C contains at least three edges as the considered graph G is a simple graph). If C is not in the same connected component $C_r(x)$ with r, then for the solution x' obtained by a mutation that flips exactly one of the 1-bits in x corresponding to the edges in C, it satisfies the claimed conditions. Thus the lemma holds.

The following discussion for the situation that C is in the same connected component $C_r(x)$ with r is divided into two cases.

(1). All vertices in C have distance not greater than 1 to r. There exists an edge e in C whose endpoints are not r. Moreover, for any vertex $v \in V$, the shortest path connecting v and r in $G(x)$ cannot contain the edge e. Thus for the solution obtained by the mutation that flips the 1-bit corresponding to the edge e in x and nothing else, it satisfies the claimed conditions.

(2). There exists a vertex v in C whose distance to r is not less than 2. Let P be a shortest path connecting v and r in $G(x)$, and v_1 be the neighbor of v that is in C, but not in P. Denote by x' the solution obtained by the mutation that flips the 1-bit corresponding to $[v, v_1]$ in x and nothing else. Note that v_1 cannot be the root r. In the following discussion, we show that $N_{d>2}(x) = N_{d>2}(x')$. Firstly, as $G(x')$ is a subgraph of $G(x)$, we have the observation that

$$N_{d>2}(x) \leq N_{d>2}(x'). \tag{3}$$

Now we assume that there exists a vertex $v' \in V \setminus \{v\}$ with $d_{G(x)}(v', r) \leq 2$, but $d_{G(x')}(v', r) > 2$. Then we have that any shortest path connecting v' and r in $G(x)$ always contains the edge $[v, v_1]$, implying that $d_{G(x)}(v', r) \geq 3$, a contradiction to the assumption. Thus $N_{2\geq d>0}(x) \leq N_{2\geq d>0}(x')$, implying

$$N_{d>2}(x) \geq N_{d>2}(x'). \tag{4}$$

By Inequalities (3) and (4), we have $N_{d>2}(x) = N_{d>2}(x')$. Combining the equality with the fact that $Cost(x) > Cost(x')$ and $N_{cc}(x) = N_{cc}(x')$, solution x' satisfies the claimed conditions. □

4.1 (1+1) EA

In this subsection, we study the performance of the (1+1) EA with respect to eight different fitness functions separately.

THEOREM 4.2. *The expected time of the (1+1) EA with fitness function f_1 to obtain a feasible solution of DR-(1,2)-MSTP on G is $O(m^2 \log n)$.*

PROOF. We first consider the case that the maintained solution x has $f_1(x) \geq m^2$, i.e., $N_{cc}(x) > 1$ or $N_{cc}(x) + |x|_1 > n + 1$.

(1). $N_{cc}(x) = c > 1$. Then there are at least $\binom{c}{2}$ edges in $E \setminus E(x)$, each of whose inclusion results in a new solution x' with $N_{cc}(x') = c - 1$ and $N_{cc}(x') + |x'|_1 = N_{cc}(x) + |x|_1$, which can be accepted by the algorithm. The mutation flipping exactly one of the 0-bits in x

corresponding to the $\binom{c}{2}$ edges and nothing else can be generated with probability $\Omega(\frac{c^2}{e \cdot m})$.

(2). $N_{cc}(x)+|x|_1 > n+1$. Then there are at least $N_{cc}(x)+|x|_1-n-1$ edges in $E(x)$, each of whose removal results in a new solution x' with $N_{cc}(x') = N_{cc}(x)$ and $|x'|_1 < |x|_1$, which can be accepted by the algorithm. The mutation that flips exactly one of the 1-bits in x corresponding to the $N_{cc}(x) + |x|_1 - n - 1$ edges and nothing else can be generated with probability $\Omega(\frac{N_{cc}(x)+|x|_1-n-1}{e \cdot m})$.

The above analysis gives that the algorithm takes expected time $O(m)$ to obtain a solution x' with

$$2N_{cc}(x') + |x'|_1 - n - 2 < 2N_{cc}(x) + |x|_1 - n - 2,$$

which can be accepted. As $2N_{cc}(0^m) + |0^m|_1 - n - 2 = n$, $2N_{cc}(x) + |x|_1 - n - 2 \leq n$, implying that the algorithm takes expected time $O(mn)$ to obtain a solution x_1 with $|x_1|_1 = n$ and $N_{cc}(x_1) = 1$.

However, $N_{d>2}(x_1)$ may be greater than 0, i.e., $E_d(x_1) \neq \emptyset$. Thus in the following discussion, we assume that there exists an edge $[v, p] \in E_d(x_1)$, where p is the neighbor of v that is in the shortest path connecting v and r in $G(x_1)$. Note that no solution x' with $N_{cc}(x') > 1$ or $|x'|_1 \neq n$ can be accepted ever again as $f_1(x') > f_1(x_1)$. If the edge $[v, p]$ in $G(x_1)$ is replaced with the edge $[v, r]$, then a new solution x' with $N_{d>2}(x') < N_{d>2}(x_1)$ is constructed, which can be accepted by the algorithm. The mutation that swaps the two bits corresponding to the two edges $[v, p]$ and $[v, r]$ in x can be generated with probability $\Omega(\frac{1}{e \cdot m^2})$. Considering all edges in $E_d(x_1)$, the algorithm takes expected time $O(m^2/|E_d(x_1)|)$ to get such an improved solution. Observe that $N_{d>2}(x_1)$ can be upper bounded by $n-2$, i.e., $|E_d(x_1)| \leq n-2$. Thus the expected time of the algorithm to get a solution x_2 with $N_{d>2}(x_2) = 0$, $N_{cc}(x_2) = 1$, and $|x_2|_1 = n$, starting with the solution x_1 can be bounded by

$$O\left(\sum_{i=1}^{n-2} \frac{m^2}{i}\right) = O(m^2 \log n).$$

Summarizing the above analysis, the (1+1) EA with f_1 takes expected time $O(m^2 \log n)$ to obtain a feasible solution of G. □

The proof given in Theorem 4.2 for the (1+1) EA with fitness function f_1 applies to the (1+1) EA with fitness function f_1', where f_1 considers the term $N_{d>2}(x)$, but f_1' considers the term $N_{n \geq d>2}(x)$. Thus we have the following theorem.

Theorem 4.3. *The expected time of the* (1+1) *EA with fitness function* f_1' *to obtain a feasible solution of DR-(1,2)-MSTP on G is* $O(m^2 \log n)$.

Now we consider another setting of the scales of penalties, more specifically, the scale of the penalty on the presence of vertices whose distances to r is greater than 2 is larger than that on the disconnectivity of the graph $G(x)$.

Theorem 4.4. *The expected time of the* (1+1) *EA with fitness function* f_2 *to obtain a feasible solution of DR-(1,2)-MSTP on G is* $O(m^2 \log n)$.

Proof. The reasoning runs in a similar way to that of Theorem 4.2. We first consider the case that the maintained solution x has $f_2(x) \geq m^2$, i.e., $N_{d>2}(x) > 0$ or $N_{cc}(x) + |x|_1 > n + 1$.

(1). $N_{d>2}(x) > 0$. Let v be an arbitrary vertex with $d_{G(x)}(v, r) > 2$. If v is not in the same connected component with r in $G(x)$, then the mutation flipping the 0-bit corresponding to the edge $[v, r]$ in x and nothing else results in a new solution x' with $N_{d>2}(x') < N_{d>2}(x)$ and $N_{cc}(x') < N_{cc}(x)$, which can be accepted by the algorithm. If v is in the same connected component with r in $G(x)$, then let p be the neighbor of v that is in a shortest path connecting r and v in $G(x)$. The mutation that swaps the two bits corresponding to the edges $[v, p]$ and $[v, r]$ in x constructs a new solution x' with $N_{d>2}(x') < N_{d>2}(x)$ and $N_{cc}(x') + |x'|_1 = N_{cc}(x) + |x|_1$, which can be accepted by the algorithm. Thus under the case that $N_{d>2}(x) > 0$, the algorithm takes expected time $O(m^2/N_{d>2}(x))$ to get an accepted solution x' with $N_{d>2}(x') < N_{d>2}(x)$.

(2). $N_{cc}(x) + |x|_1 > n + 1$. For a cycle C in $G(x)$, the removal of an edge in C results in a new solution x' with $N_{cc}(x') = N_{cc}(x)$ and $|x'|_1 < |x|_1$. However, $N_{d>2}(x')$ may be greater than $N_{d>2}(x)$. Thus we have to remove the extra edges in $G(x)$ carefully. By Lemma 4.1, there are $N_{cc}(x) + |x|_1 - n - 1$ 1-bits in x, each of whose flip results in a solution x' with $Cost(x') < Cost(x)$, $|x'|_1 = |x|_1 - 1$, $N_{cc}(x') = N_{cc}(x)$, and $N_{d>2}(x') = N_{d>2}(x)$, which can be accepted by the algorithm. Thus the algorithm takes expected time $O(m/(N_{cc}(x) + |x|_1 - n - 1))$ to get such an accepted solution.

Summarizing the above analysis, if $N_{d>2}(x) + (N_{cc}(x) + |x|_1 - n - 1) > 0$, then the algorithm takes expected time

$$O\left(\frac{m^2}{\max\{N_{d>2}(x), N_{cc}(x) + |x|_1 - n - 1\}}\right) = O\left(\frac{m^2}{N_{d>2}(x) + N_{cc}(x) + |x|_1 - n - 1}\right)$$

to get a solution x' with

$$N_{d>2}(x') + N_{cc}(x') + |x'|_1 < N_{d>2}(x) + N_{cc}(x) + |x|_1.$$

Observe that $N_{d>2}(x) + N_{cc}(x) + |x|_1 - n - 1$ can be upper bounded by n (as $f_2(0^m) = 4m^2 n$). Thus considering all possible values of $N_{d>2}(x) + N_{cc}(x) + |x|_1 - n - 1$ and the corresponding waiting time, the algorithm takes expected time

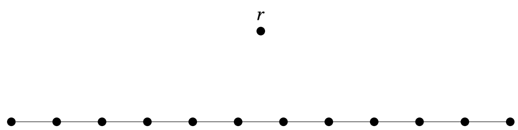
$$O\left(\sum_{i=1}^{n} \frac{m^2}{i}\right) = O(m^2 \log n)$$

to get a solution x_1 with $N_{d>2}(x_1) = 0$ and $N_{cc}(x_1) + |x_1|_1 = n + 1$, implying that x_1 is a feasible solution. □

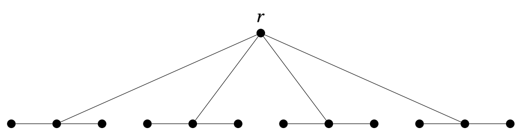
According to the reasoning given in Theorems 4.2 and 4.4, it is not hard to get the following theorem for fitness function f_3, which consider the setting: the scale of the penalty on the presence of vertices whose distances to r are greater than 2 is the same as that on the disconnectivity of the graph $G(x)$.

Theorem 4.5. *The expected time of the* (1+1) *EA with fitness function* f_3 *to obtain a feasible solution of DR-(1,2)-MSTP on G is* $O(m^2 \log n)$.

For the (1+1) EA with fitness function f_2', we first construct a special weight function W_s for G and a solution x_s that the algorithm may maintain (illustrated in Figure 1a). Then we show that once the algorithm maintains the solution x_s, then it needs exponential expected time to get an improved solution to G with weight function W_s.



(a) the graph $G(x_s)$ corresponding to the special solution x_s.



(b) the optimal approach for the bad case.

Figure 1: A bad situation for the (1+1) EA with fitness function f'_2. All edges given in Figure 1a have weight 1, and the other edges (recall that G is complete) have weight 2, with respect to the weight function W_s.

Theorem 4.6. *The (1+1) EA with fitness function f'_2 takes expected time $\Omega(m^{2\lceil \frac{n}{3} \rceil - 1})$ to obtain a feasible solution of DR-(1,2)-MSTP on G with weight function W_s, once it maintains the solution x_s.*

Proof. As $f'_2(0^m) = 2mn$, $N_{n \geq d > 2}(x) = 0$ and $N_{\text{cc}}(x) + |x|_1 = n + 1$ hold for any solution x accepted by the (1+1) EA with fitness function f'_2. Assume that the algorithm maintains the solution x_s (illustrated in Figure 1a), where $G(x_s)$ contains two connected components: one is the isolated vertex r, the other one C is a path connecting all vertices in V with $n - 1$ edges. Observe that the solution x_s is the best solution with Hamming weight $n - 1$ for G with weight function W_s, with respect to the fitness function f'_2. The unique way to get an improved solution compared to x_s is to decrease the number of connected components in the solution. However, because of the path structure of C, no matter which edge in $E \setminus E(x_s)$ is added into $G(x_s)$ to connect r with C, it always causes a vertex v with $d_{G(x')}(v, r) > 2$, i.e., $N_{n \geq d > 2}(x') > 0$, where x' is the resulting offspring. Thus the solution x' would be rejected.

The optimal approach that flips the minimum number of bits in x_s to get an improved solution is illustrated as Figure 1b. Observe that it still needs flipping $2\lceil \frac{n}{3} \rceil - 1$ edges at the same time, implying that the algorithm needs expected time $\Omega(m^{2\lceil \frac{n}{3} \rceil - 1})$. □

The (1+1) EA with fitness function f'_3 may get struck in the same situation as that given in Figure 1a, thus we have the following theorem for it.

Theorem 4.7. *The (1+1) EA with fitness function f'_3 takes expected time $\Omega(m^{2\lceil \frac{n}{3} \rceil - 1})$ to obtain a feasible solution of DR-(1,2)-MSTP on G with weight function W_s, once it maintains the solution x_s.*

The discussion for Theorems 4.3, 4.6, and 4.7 shows that if we only consider the vertices that are in the same connected component with r and their distances to r, then we may lose the control of the connected components that do not contain r, and get stuck in a local solution from which the algorithm may need exponential expected time to get an improved solution. Meanwhile, it shows the importance of the setting of penalties on the considered factors to the performance of the algorithm.

The following theorem considers the performance of the (1+1) EA with fitness function f_4 that has a penalty on the term $N^{\text{nc}}_{\text{cc}}(x)$, where $N^{\text{nc}}_{\text{cc}}(x)$ counts the number of non-claw connected components in $G(x)$, except the one $C_r(x)$ containing r.

Theorem 4.8. *The expected time of the (1+1) EA with fitness function f_4 to obtain a feasible solution of DR-(1,2)-MSTP on G is $O(m \log n)$.*

Proof. Let x be the solution maintained by the (1+1) EA with fitness function f_4. Since $f_4(0^m) = 2mn$, all connected components in $G(x)$ are claw components except the one $C_r(x)$ containing the vertex r. Moreover, $N_{n \geq d > 2}(x) = 0$. The thing remaining to be done is connecting all these connected components together, in a feasible way. Let C be a claw component in $G(x)$ except $C_r(x)$.

If C is a singleton component, then the feasible way is to connect the vertex in C to r, or a child of r (if it has), or the center of another connected component. If C is not a singleton component, then the unique feasible way is to connect its center to r. That is because if a leaf or the center of C is connected to a vertex (except r) in C_r, then there exists a vertex in C whose distance to r would be between 3 and n, implying the obtained solution cannot be accepted. If C is connected to another component C' that has more than one vertex, then the resulting connected component is not a claw component, implying the obtained solution cannot be accepted as well.

Summarizing the above analysis, connecting the center of C to r is always feasible. Thus for the $N_{\text{cc}}(x) = c$ connected components in $G(x)$, there are at least $c-1$ edges, each of whose inclusion results in a solution with improved fitness. The mutation flipping exactly one of the 0-bits corresponding to these edges in x and nothing else takes expected time $O(\frac{m}{c-1})$. Summing over the waiting time for all values $2 \leq c \leq n + 1$, we have that the algorithm takes expected time $O(m \log n)$ to obtain a feasible solution. □

The following theorem considers the performance of the (1+1) EA with fitness function f_5 that has a penalty on the term $N^{>1}_{\text{cc}}(x)$, where $N^{>1}_{\text{cc}}(x)$ counts the number of connected components with more than one vertex in $G(x)$, except the one $C_r(x)$.

Theorem 4.9. *The expected time of the (1+1) EA with fitness function f_5 to obtain a feasible solution of DR-(1,2)-MSTP on G is $O(m \log n)$.*

Proof. Let x be the solution maintained by the algorithm. Since $f_5(0^m) = 2mn$, we have that $N^{>1}_{\text{cc}}(x) = 0$, i.e., except the connected component $C_r(x)$ containing r, all the other connected components in $G(x)$ are singleton components. Moreover, we have that $N_{n \geq d > 2}(x) = 0$ and $N_{\text{cc}}(x) + |x|_1 = n + 1$.

The feasible way to decrease the number of connected components in $G(x)$ is connecting a singleton component to $C_r(x)$, more specifically, to either the vertex r or a child of r (if it has) in $C_r(x)$. Thus for the $N_{\text{cc}}(x) = c$ connected components in $G(x)$, there are at least $c - 1$ edges, each of whose inclusion results in a solution x' with $N_{\text{cc}}(x') = N_{\text{cc}}(x) - 1$ that can be accepted by the algorithm. The mutation flipping exactly one of the 0-bits corresponding to these edges in x and nothing else can be generated with probability $\Omega(\frac{c-1}{m})$, i.e., the algorithm takes expected time $O(\frac{m}{c-1})$ to get such an improved solution. Summing over the waiting time for all values

$2 \leq c \leq n+1$, the algorithm takes expected time $O(m \log n)$ to obtain a feasible solution. □

4.2 MOEA and Its Variants

In this subsection, we study the performance of the MOEA and its two variants for DR-(1,2)-MSTP separately.

Theorem 4.10. *The expected time of the MOEA to obtain a feasible solution of DR-(1,2)-MSTP on G is $O(mn \log n)$.*

Proof. Given a solution x of DR-(1,2)-MSTP on G, if $N_{\text{cc}}(x) + |x|_1 = n+1$ and $N_{d>2}(x) + 1 = N_{\text{cc}}(x)$, then it is a *candidate*. That is, all the vertices that are in the same connected component $C_r(x)$ with r have distance at most 2 to r, and all the other vertices that are not in $C_r(x)$ are singleton components. By f_{MOEA} and the definition of dominance with respect to f_{MOEA}, we have that a candidate x in the population S maintained by the algorithm can only be replaced by another candidate x' such that $|x'|_1 = |x|_1$ and $Cost(x') \leq Cost(x)$.

Let x_1 be the candidate in S with the maximum Hamming weight, where $|x_1|_1$ is assumed to be less than n. Let v be an arbitrary vertex in V that is a singleton component in $G(x_1)$. Including the edge $[v, r]$ into x constructs a new candidate x_2 with Hamming weight $|x_1|_1 + 1$. Note that the candidate x_2 would be accepted by the algorithm; otherwise, the population contains another candidate x_2' such that $|x_2'|_1 = |x_2|_1$ and $Cost(x_2') < Cost(x_2)$, a contradiction to that x_1 is the candidate in S with the maximum Hamming weight.

Since the size of the population S is bounded by $n+1$, the mutation choosing the candidate x_1 and flipping one of the 0-bits corresponding to the $N_{\text{cc}}(x_1) - 1$ edges between r and the $N_{\text{cc}}(x_1) - 1$ singleton components in $G(x_1)$ is generated with probability $\Omega(\frac{1}{n+1} \cdot \frac{N_{\text{cc}}(x_1)-1}{e \cdot m})$. Hence the algorithm takes expected time $O(\frac{mn}{N_{\text{cc}}(x_1)-1}) = O(\frac{mn}{n-|x|_1})$ to get the candidate x_2, improving the maximum value of the Hamming weights of the candidates in S by 1. Combining the above conclusion and the fact that 0^m is a candidate with Hamming weight 0, summing over the waiting time for all values $0 \leq |x_1|_1 \leq n-1$ gives that the algorithm takes expected time $O(mn \log n)$ to get a feasible solution. □

Theorem 4.11. *The expected time of the MOEA-1 to obtain a feasible solution of DR-(1,2)-MSTP on G is $O(mn)$.*

Proof. We first consider the expected time of the MOEA-1 to find a solution with Hamming weight n. Let x be a solution in the maintained population S with $0 \leq |x|_1 < n$. Flipping exactly one 0-bit in x and nothing else results in a new solution x' that dominates x with respect to $\succcurlyeq_{\text{MOEA-1}}$, thus the algorithm takes expected time $O(m/(m - |x|_1))$ to obtain a solution that has Hamming weight $|x|_1 + 1$ (recall that the population maintained by the MOEA-1 has size at most 2, hence the solution is chosen with probability $\Omega(1)$). Consider the potential of the population maintained by the MOEA-1, which is defined as the maximum number of 0-bits that the solutions in the population have. The Multiplicative Drift Theorem [10] gives that the algorithm takes expected time $O(m \ln \frac{m}{m-n})$ to obtain a population with potential $m - n$, which contains a solution with Hamming weight n.

Now we assume that the population S contains a solution x_n with $|x_n|_1 = n$, and consider the expected time of the algorithm to find a feasible solution starting with x_n. If $N_{d>2}(x_n) = 0$, then x_n is feasible, and the proof is done. Thus in the following discussion, we assume that x_n is infeasible.

If $N_{\text{cc}}(x_n) > 1$, then let v be a vertex that is not in the connected component $C_r(x_n)$; otherwise, let v be one of the vertices that have distance larger than 2 to r in $G(x_n)$. Let x_{n+1}' be the solution obtained by including the edge $[v, r]$ into x_n. Observe that

$$N_{\text{cc}}(x_{n+1}') + N_{d>2}(x_{n+1}') < N_{\text{cc}}(x_n) + N_{d>2}(x_n),$$

and the weight on $[v, r]$ is at most 2, thus

$$f_{\text{M}}^2(x_{n+1}') < f_{\text{M}}^2(x_n) - (2m-2).$$

The algorithm generates the mutation that flips the 0-bit corresponding to the edge $[v, r]$ in x_n and nothing else with probability $\Omega(1/m)$, i.e., taking expected time $O(m)$ to get x_{n+1}'.

If the population S has no solution with Hamming weight $n+1$, or $f_{\text{M}}^2(x_{n+1}') \leq f_{\text{M}}^2(x_{n+1})$ for the solution x_{n+1} maintained in the population with Hamming weight $n+1$, then the solution x_{n+1}' would be accepted; otherwise, rejected. Thus in the following discussion, we assume that the population contains a solution x_{n+1}'' with Hamming weight $n+1$ such that $f_{\text{M}}^2(x_{n+1}'') \leq f_{\text{M}}^2(x_{n+1}')$.

By Lemma 4.1, we can get a mutation on x_{n+1}'' that results in a solution x_n' with Hamming weight n such that $f_{\text{M}}^2(x_n') < f_{\text{M}}^2(x_{n+1}'')$, where the generation of the mutation takes expected time $O(m)$. As

$$f_{\text{M}}^2(x_n') < f_{\text{M}}^2(x_{n+1}'') \leq f_{\text{M}}^2(x_{n+1}') < f_{\text{M}}^2(x_n) - (2m-2),$$

Thus the solution x_n' would be accepted, replacing x_n.

Considering the mutation generating x_{n+1}' and the one generating x_n', the algorithm takes expected time $O(m)$ to get the improved solution x_n' with Hamming weight n such that

$$f_{\text{M}}^2(x_n') \leq f_{\text{M}}^2(x_n) - 2m.$$

Since the value of $f_{\text{M}}^2(x_n)$ can be upper bounded by $4mn + 2n$, the Additive Drift Theorem [15] gives that the algorithm takes expected time $O(mn)$ to get a solution x^* with $|x^*|_1 = n$ and $f_{\text{M}}^2(x^*) < 2m$ (i.e., x^* is a feasible solution) starting with the solution x_n. Combining the time with the expected time to get x_n starting with 0^m, we have the claimed expected time for the MOEA-1. □

Theorem 4.12. *The expected time of the MOEA-2 to obtain a feasible solution of DR-(1,2)-MSTP on G is $O(m \log n)$.*

Proof. Let x the solution in the population maintained by the algorithm with $|E_d(x)| = 0$ (note that the population always has such a solution as $|E_d(0^m)| = 0$). Since $f_{\text{M2}}^2(0^m) = 2mn$, the solution x has $N_{\text{cc}}(x) + |x|_1 = n+1$, and $N_{\text{cc}}^{>1}(x) = 0$. If $f_{\text{M2}}^2(x) \geq 2m$, then $N_{\text{cc}}(x) > 1$. As $N_{\text{cc}}^{>1}(x) = 0$, all connected components in $G(x)$ except $C_r(x)$ are singleton components. Let v be an arbitrary vertex that is a singleton component in $G(x)$. The mutation flipping exactly the 0-bit corresponding the edge $[v, r]$ in x and nothing else results in a solution that dominates x with respect to f_{M2}, which can be generated with probability $\Omega(1/m)$. Considering the population size of the algorithm and all singleton components in $G(x)$, the expected time to get a solution x' with $|E_d(x')| = 0$ and $N_{\text{cc}}(x') < N_{\text{cc}}(x)$ is $O(m/(N_{\text{cc}}(x) - 1))$.

Observe that $N_{\text{cc}}(x)$ is at most $n+1$. Thus the expected time of the algorithm to get a solution x^* with $|E_d(x^*)| = 0$ and $N_{\text{cc}}(x^*) = 1$

can be bounded by

$$O\left(\sum_{i=1}^{n}\frac{m}{i}\right)=O(m\log n).$$

Remark that the solution x^* also satisfies $N_{cc}(x^*)+|x^*|_1=n+1$, thus x^* is a feasible solution. □

The above proof for Theorem 4.12 is based on the solution x with $|E_d(x)|=0$ in the population. For the other possible solution x' with $|E_d(x')|=1$ in the population, we will show its power in the next section.

5 IMPROVED ANALYSIS OF DR-(1,2)-MSTP

The section studies the improved performance of the four algorithms based on several local search operations.

THEOREM 5.1. *The expected time of the* (1+1) *EA with one of the fitness functions* f_1-f_5 *and* f_1'*, to obtain a* $\frac{3}{2}$*-approximate solution of DR-(1,2)-MSTP on* G *is* $O(m^6 n)$*.*

PROOF. Assume that the algorithm has obtained a feasible solution x_1. By Theorems 4.2, 4.3, 4.4, 4.5, 4.8, and 4.9, the expected time of the algorithm to obtain x_1 can be bounded by $O(m^2\log n)$. In the following, we give several operations to optimize x_1.

We first give some related notions below. As $G(x_1)$ is a spanning tree of G, and the specific vertex r is treated as the root of the tree, there is a well-defined ancestor-descendant relationship in $G(x_1)$. More specifically, given an edge $[v,v']$ with two endpoints v and v' in $G(x_1)$, if v is in the unique path connecting v' and r, then v is the *parent* of v', and v' is a *child* of v. Similarly, if the vertex v'' is a child of v', then v'' is a *grandchild* of v in $G(x_1)$.

Operation 1. If there is a grandchild v_1 of r in $G(x_1)$ such that the edge between v_1 and its parent p_1 in $G(x_1)$ has weight 2, but $W([v_1,r])=1$, then swap the edge $[v_1,p_1]$ with the edge $[v_1,r]$. The illustrations of Operation 1 and the following four operations are given in Figure 2.

Operation 2. If there is a child v_1 and a grandchild v_2 of r in $G(x_1)$ such that the edge between v_2 and its parent p_2 (not v_1) has weight 2, but $W([v_1,v_2])=1$, then swap $[v_2,p_2]$ with $[v_1,v_2]$.

Operation 3. If there are two children v_1 and v_2 of r in $G(x_1)$ such that v_1 has no child, and $W([v_1,r])=2$ and $W([v_1,v_2])=1$, then swap the edge $[v_1,r]$ with the edge $[v_1,v_2]$.

Obviously, each application of Operation 1, Operation 2, and Operation 3 on x_1 gets an improved solution x_1' that can be accepted. The mutation corresponding to the application can be generated with probability $\Omega(1/m^2)$, i.e., the algorithm takes expected time $O(m^2)$ to get the improved solution x_1'.

Operation 4. If there is a grandchild v_1 of r, and a vertex v_2 that is either a grandchild or a child with no child of r in $G(x_1)$, such that the edge between v_1 and its parent p_1 in $G(x_1)$ and the edge $[v_1,r]$ have the same weight, but the edge between v_2 and its parent p_2 (may be r) in $G(x_1)$ has a larger weight than the edge $[v_1,v_2]$, then swap $[v_1,p_1]$ and $[v_2,p_2]$ with $[v_1,r]$ and $[v_1,v_2]$.

Each application of Operation 4 on x_1 gets an improved solution x_1', which can be accepted. The mutation corresponding to the application can be generated with probability $\Omega(1/m^4)$, i.e., the algorithm takes expected time $O(m^4)$ to get x_1'. Now we consider a grandchild v_1 of r, and two vertices v_2 and v_3, each of which is either a grandchild of r or a child of r with no child in $G(x_1)$. Let p_1, p_2, and p_3 be the parents of v_1, v_2, and v_3, respectively (p_2 and p_3 may be r if v_2 and v_3 are the children of r in $G(x_1)$).

Operation 5. If $W([v_1,p_1])=1=W([v_1,v_2])=W([v_1,v_3])$, but $W([v_1,r])=2=W([v_2,p_2])=W([v_3,p_3])$, then swap the edges $[v_1,p_1]$, $[v_2,p_2]$, $[v_3,p_3]$ with $[v_1,r]$, $[v_1,v_2]$, and $[v_1,v_3]$.

Each application of Operation 5 gets an improved solution x_1' that can be accepted. The mutation corresponding to the application can be generated with probability $\Omega(1/m^6)$, i.e., the algorithm takes expected time $O(m^6)$ to get x_1'.

Since $Cost(x_1)\le 2n$, and $Cost(x^*)\ge n$, where x^* is an optimal solution for DR-(1,2)-MSTP on G, Operation 1-5 can be applied at most n times. That is, starting with x_1, the algorithm takes expected time $O(m^6 n)$ to get a feasible solution x_2 on which Operation 1-5 are not applicable.

Now we analyze the cost of the solution x_2, $Cost(x_2)$. First of all, we partition the vertices of V into the following subsets according to the structure of $G(x_2)$.

1) $V_{11}(x_2)$, contains all the vertices $v\in V$, where v is the child of r in $G(x_2)$, and $W([v,r])=1$;
2) $V_{12}(x_2)$, contains all the vertices $v\in V$, where v is the child of r in $G(x_2)$, and $W([v,r])=2$;
3) $V_{21}(x_2)$, contains all the vertices $v\in V$, where v is the grandchild of r in $G(x_2)$, and $W([v,p])=1$ (p is the parent of v in $G(x_2)$);
4) $V_{22}(x_2)$, contains all the vertices $v\in V$, where v is the grandchild of r in $G(x_2)$, and $W([v,p])=2$ (p is the parent of v in $G(x_2)$).

Moreover, the vertices in $V_{12}(x_2)$ are partitioned into the following two subsets.

1) $V_{12}^0(x_2)$, contains all the vertices $v\in V_{12}(x_2)$, where v has no children in $G(x_2)$;
2) $V_{12}^1(x_2)$, contains all the vertices $v\in V_{12}(x_2)$, where v has at least one child in $G(x_2)$.

Consider a vertex $v\in V_{22}(x_2)\cup V_{12}^0(x_2)$. Firstly, $W([v,v'])=2$ for any vertex $v'\in\{r\}\cup V_{11}(x_2)\cup V_{12}(x_2)\setminus\{v\}$; otherwise, Operation 1 or 2 or 3 is applicable on x_2. Then, $W([v,v'])=2$ for any vertex $v'\in V_{22}(x_2)\setminus\{v\}$; otherwise, Operation 4 is applicable on x_2. Thus, $W([v,v'])=2$ for any vertex $v'\in\{r\}\cup V_{11}(x_2)\cup V_{12}(x_2)\cup V_{22}(x_2)\setminus\{v\}$. If there exists a vertex $v'\in V_{21}(x_2)$ such that $W([v,v'])=1$, then $W([v',p])=1$ and $W([v',r])=2$, where p is the parent of v' in $G(x_2)$; otherwise, Operation 4 is applicable on x_2.

Let v_1 be a vertex in $V_{22}(x_2)\cup V_{12}^0(x_2)$ with $N_1(v_1)\ne\emptyset$ (recall that N_1 denotes the set containing all the vertices v' in G with $W([v_1,v'])=1$; the above analysis gives that $N_1(v_1)\subset V_{21}(x_2)$). If the parent p_1 of v_1 in $G(x^*)$ is a vertex of $\{r\}\cup V\setminus N_1(v_1)$ (recall that x^* is an optimal solution of DR-(1,2)-MSTP on G), then the edge between v_1 and p_1 in $G(x^*)$ has weight 2; otherwise, by the above analysis, the edge between p_1 and its parent r in $G(x^*)$ has weight 2.

Assume that the parent p_1 of v_1 in $G(x^*)$ is a vertex of $N_1(v_1)$, where $N_1(v_1)\subset V_{21}(x_2)$. We have that if there is a vertex v_2 in $V_{22}(x_2)\cup V_{12}^0(x_2)\setminus\{v_1\}$ that is also the child of p_1 in $G(x^*)$, then $W([p_1,v_2])=2$; otherwise, Operation 5 is applicable on $G(x_2)$, with respect to v_1, v_2, and p_1. That is, if there is a subset $V'\subset(V_{22}(x_2)\cup V_{12}^0(x_2))$ in which the vertices have the same common parent p in $G(x^*)$, then one of the following three cases holds.

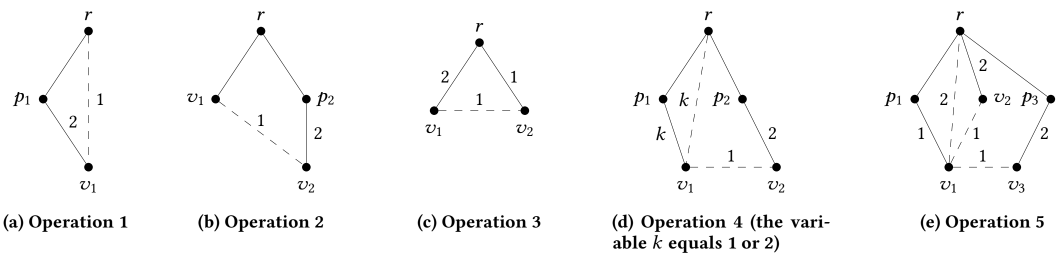


Figure 2: Illustrations of Operation 1-5 considered in Theorem 5.1. In particular, the illustration of Operation 4 given above only considers the case that v_2 is a grandchild of r in $G(x_1)$, and that of Operation 5 given above only considers the case that v_2 is a child of r with no children, and v_3 is a grandchild of r in $G(x_1)$.

1) p is the vertex r, thus all the edges between p and the vertices in V' have weight 2;
2) p is a child of r, and there is a vertex $v \in V'$ with $p \in N_1(v)$, then all edges between p and the vertices in $V' \setminus \{v\}$ have weight 2, the edge $[v, p]$ has weight 1, and the edge $[p, r]$ has weight 2;
3) p is a child of r, and there is no vertex $v \in V'$ with $p \in N_1(v)$, then all edges between p and the vertices in V' have weight 2.

Summarizing the above analysis, we have

$$Cost(x^*) \quad \geq \quad n + |V_{22}(x_2)| + |V_{12}^0(x_2)|.$$

The following inequality can be easily derived, where the last inequality relation holds because each vertex in $V_{12}^1(x_2)$ has at least one child in $G(x_2)$, i.e., $|V_{12}^1(x_2)| \leq n/2$.

$$\begin{aligned} Cost(x_2) &= |V_{11}(x_2)| + 2|V_{12}(x_2)| + |V_{21}(x_2)| + 2|V_{22}(x_2)| \\ &= n + |V_{12}(x_2)| + |V_{22}(x_2)| \\ &= n + |V_{12}^0(x_2)| + |V_{12}^1(x_2)| + |V_{22}(x_2)| \\ &\leq \frac{3n}{2} + |V_{12}^0(x_2)| + |V_{22}(x_2)|. \end{aligned}$$

Therefore, the approximate ratio of x_2 is

$$\frac{Cost(x_2)}{Cost(x^*)} \leq \frac{\frac{3n}{2} + |V_{12}^0(x_2)| + |V_{22}(x_2)|}{n + |V_{12}^0(x_2)| + |V_{22}(x_2)|} \leq \frac{3}{2}.$$

The above conclusion gives that the algorithm takes expected time $O(m^6 n)$ to obtain a solution with ratio $\frac{3}{2}$. □

By almost the same reasoning given in the proof for Theorem 5.1, and the sizes of the populations maintained by the MOEA and MOEA-1, we can get the following theorem.

Theorem 5.2. *The expected time of the MOEA and MOEA-1 to obtain a $\frac{3}{2}$-approximate solution of DR-(1,2)-MSTP on G is $O(m^6 n^2)$ and $O(m^6 n)$, respectively.*

Now we consider the improved performance of the MOEA-2 based on the first four local search operations given in the proof for Theorem 5.1, and a new designed local search operation.

Theorem 5.3. *The expected time of the MOEA-2 to obtain a $\frac{3}{2}$-approximate solution of DR-(1,2)-MSTP on G is $O(m^4 n)$.*

Proof. Assume that the algorithm has obtained a population that contains a feasible solution x_0, i.e., $|E_d(x_0)| = 0$. By Theorem 4.12, the algorithm takes expected time $O(m \log n)$ to get such a population. In the following, we give several operations to optimize x_0. Firstly, we adopt the Operations 1-4 given in the proof of Theorem 5.1, then present a new operation as follows.

Consider a grandchild v_1 of r, and two vertices v_2 and v_3, each of which is either a grandchild of r or a child of r with no child in $G(x_0)$. Let p_2 and p_3 be the parents of v_2 and v_3, respectively (p_2 and p_3 may be r if v_2 and v_3 are the children of r in $G(x_0)$).

Operation 6. If $W([v_1, v_2]) = W([v_1, v_3]) = 1$, but $W([v_2, p_2]) = W([v_3, p_3]) = 2$, then swap the edges $[v_2, p_2]$, $[v_3, p_3]$ with $[v_1, v_2]$, and $[v_1, v_3]$.

Denote by x' the solution obtained by Operation 6 on x_0. Observe that $|E_d(x')| = 1$ and $f_{M2}^2(x') = f_{M2}^2(x_0) - 2$. If x' cannot be accepted by the algorithm, then its population contains a solution x_1 with $|E_d(x_1)| = 1$ satisfying $f_{M2}^2(x_1) < f_{M2}^2(x')$. Thus in the following, we assume that the population contains a solution x_1 with $|E_d(x_1)| = 1$ satisfying $f_{M2}^2(x_1) \leq f_{M2}^2(x')$. Denote by $[v, p]$ the unique edge in $E_d(x_1)$, where p is the parent of v in $G(x_1)$. Swapping the edge $[v, p]$ with the edge $[v, r]$, we can get a solution x'' with $|E_d(x'')| = 0$. As the edges $[v, p]$ and $[v, r]$ may have weight 1 and 2, respectively, $f_{M2}^2(x'') \leq f_{M2}^2(x_1) + 1$. Therefore,

$$f_{M2}^2(x'') \leq f_{M2}^2(x_1) + 1 \leq f_{M2}^2(x') + 1 \leq f_{M2}^2(x_0) - 1.$$

Summarizing the above analysis, Operation 6 and the subsequent operation can improve the feasible solution in the population by at least one with respect to f_{M2}^2. The mutations corresponding to the two operations can be generated with probability $\Omega(1/m^4)$ and $\Omega(1/m^2)$, respectively, i.e., they take expected time $O(m^4)$.

Since $Cost(x_0) \leq 2n$, and $Cost(x^*) \geq n$, where x^* is an optimal solution for DR-(1,2)-MSTP on G, Operation 1-4 and 6 can be applied at most n times. That is, starting with x_1, the algorithm takes expected time $O(m^4 n)$ to get a feasible solution x_2 on which Operation 1-4 and 6 are not applicable. By the reasoning given in the proof for Theorem 5.1, the approximate ratio of x_2 is $\frac{3}{2}$. □

6 REDUCTION FROM DR-(1,2)-MSTP TO SET COVER PROBLEM

Given a feasible solution x of DR-(1,2)-MSTP on G, the graph $G(x)$ corresponds to a collection of claw components, denoted by $L(x)$, and the edges connecting them to the root r. Conversely, a collection L of claw components that covers all vertices in V corresponds to a feasible solution of DR-(1,2)-MSTP on G, where the corresponding depth restricted spanning tree can be constructed by connecting the center of each claw component $C \in L$ to r with a new edge.

Thus we study the problem DR-(1,2)-MSTP from the perspective of claw components in the following discussion. Firstly, we partition the vertices in V into the three subsets given below:

1) V_1, contains all the vertices $v \in V$ with $W([v, r]) = 1$;
2) V_2, contains all the vertices $v \in V$ with $W([v, v']) = 2$ for any vertex $v' \in \{r\} \cup V \setminus \{v\}$;
3) V_{21}, contains all vertices in $V \setminus (V_1 \cup V_2)$. That is, for each vertex $v \in V_{21}$, $W([v, r]) = 2$, but there is a vertex v' with $W([v, v']) = 1$, where v' can be a vertex of $V_1 \cup V_{21}$.

The vertices in V_{21} can be further partitioned into the following two subsets:

1) V_{21}^1, contains all the vertices $v \in V_{21}$ such that there exists a vertex $v_1 \in V_1$ with $W([v, v_1]) = 1$;
2) V_{21}^2, contains all vertices in $V_{21} \setminus V_{21}^1$, i.e., for each vertex $v \in V_{21}^2$, there does not exist a vertex $v_1 \in V_1$ with $W([v, v_1]) = 1$.

Lemma 6.1. *There exists an optimal solution x^* for DR-(1,2)-MSTP on G that satisfies the following three properties:*

1) *all vertices in $V_1 \cup V_2$ are the children of r in $G(x^*)$;*
2) *all the edges with weight 2 are incident to r in $G(x^*)$;*
3) *for each vertex $v \in V_{21}^1$, it is either the child of a vertex of V_1 in $G(x^*)$, or the center of a claw component in $L(x^*)$.*

Proof. Let x be an arbitrary optimal solution for DR-(1,2)-MSTP on G. We first consider a vertex $v_1 \in V_1$. It is easy to see that the edge e_1 between v_1 and its parent in $G(x^*)$ has weight 1; otherwise, x is not optimal. Thus if v_1 is not a child of r in $G(x^*)$, then by swapping the edge e_1 with the edge between v_1 and r whose weight is also 1, we can get another optimal solution x' such that v_1 is a child of r in $G(x')$. Similar analysis applies to the vertices in V_2.

For an edge $[v_1, v_2]$ with weight 2 in $G(x)$, where v_1 is the parent of v_2, and v_1 is not r, we have that $W([v_2, r]) = 2$. Thus by swapping the edge $[v_1, v_2]$ with the edge $[v_2, r]$, we can get another optimal solution x' such that the edge with weight 2 is incident to r.

Thus by iteratively applying the operations mentioned above on the optimal solution x, we can get an optimal solution x_2 satisfying the claimed Property (1) and (2). Now we consider Property (3) on the solution x_2 in the following discussion.

For a vertex $v \in V_{21}^1$, it cannot be a child of r with no child in $G(x_2)$; otherwise, it is not optimal. Thus if it is a child of r in $G(x_2)$, then it has at least one child and is the center of the corresponding claw component in $L(x_2)$. If v is a grandchild of r in $G(x_2)$, then the edge incident to it in $G(x_2)$ has weight 1; otherwise, it is not optimal. Recall that there exists a vertex $v' \in V_1$ with $W([v, v']) = 1$ by the definition of V_{21}^1. Thus by swapping the edge incident to it with the edge $[v, v']$, we can get another optimal solution x' such that it is the child of a vertex of V_1. Therefore, by iteratively applying the operation mentioned above on the solution x_2, we can obtain an optimal solution satisfying the three claimed properties. □

Given a claw component C, if it has no edge, or all edges in C have weight 1, then it is a *unit claw component*. Let x^* be an optimal solution satisfying all properties given in Lemma 6.1. Then we have that all claw components in $L(x^*)$ are unit claw components (by the second property given in Lemma 6.1). However, a collection of unit claw components that covers all vertices in V may not correspond to an optimal solution of DR-(1,2)-MSTP on G, since there are two types of unit claw components, which are defined as follows. Consider a unit claw component C, if the edge between its center and r has weight 1, then it is a *1-unit claw component*; otherwise, it is a *2-unit claw component*.

It is easy to see that $L(x^*)$ has the minimum number of 2-unit claw components. Thus an approach to obtain an optimal solution of DR-(1,2)-MSTP on G is looking for a collection of unit claw components to cover all vertices in V such that the collection contains the minimum number of 2-unit claw components.

The critical issue remaining to be resolved is the construction of the universal set U, which contains all considered unit claw components, and the selection of the unit claw components from U to cover all vertices in V. For the construction of U, although it is feasible to include all possible unit claw components in G into U, we find that it is unnecessary to consider all unit claw components in G. In the following discussion, we give a feasible and efficient way for the construction of U.

Firstly, by Lemma 6.1 the vertices in $V_1 \cup V_2$ can be ignored (because they can be directly connected to the root r). For a vertex $v \in V_{21}^1$, it may be the child of a vertex in V_1 in $G(x^*)$ (recall that x^* is an optimal solution satisfying the properties given in Lemma 6.1), or be the center of a claw component of $L(x^*)$ with some vertices in V_{21}^2. For a vertex $v \in V_{21}^2$, it may be a child of r in $G(x^*)$, or be in a claw component whose center is a vertex of V_{21}.

Thus we only need to consider the unit claw components whose centers are the vertices of V_{21}, and the aim is to find a collection L of unit claw components to cover all vertices in V_{21}^2. The requirement of L to cover only the vertices in V_{21}^2 is because if there are some vertices in V_{21}^1 that are not covered by L, then they can be attached to the vertices in V_1 by the edges with weight 1.

Thus an instance $I_{\text{sc}} = (\mathcal{S}, V_{21}^2)$ of the Set Cover problem [7] can be constructed as follows: let $\mathcal{S}$ contain the sets S_v for all $v \in V_{21}$, where S_v is the set containing all vertices in $N_1(v) \cap V_{21}^2$. Note that the sizes of $\mathcal{S}$ and V_{21}^2 can be bounded by $|V| = n$. For the instance $I_{\text{sc}} = (\mathcal{S}, V_{21}^2)$, the above discussion gives the lemma below.

Lemma 6.2. *Given a solution S_{sc} to the instance $I_{\text{sc}} = (\mathcal{S}, V_{21}^2)$ of the Set Cover problem, then the corresponding depth restricted spanning tree has weight $n + |V_2| + |S_{\text{sc}}|$.*

Alfandari and Paschos [1] studied an approximate algorithm for DR-(1,2)-MSTP on G, from the perspective of claw components.

They first preprocessed the instance $I_{\text{sc}} = (\mathcal{S}, V_{21}^2)$ as follows. Initialize two sets $\mathcal{S}' = \mathcal{S}$ and $C = \emptyset$ (C is to contain the covered vertices). If there exists a $S_v \in \mathcal{S}'$ with $|S_v| \geq 4$, then let $C = C \cup S_v$, $\mathcal{S}' = \mathcal{S}' \setminus \{S_v\}$, and $S = S \setminus C$ for all $S \in \mathcal{S}'$. Iteratively apply the above operation until $|S| \leq 3$ for any $S \in \mathcal{S}'$. Then a new instance $I_{3\text{sc}} = (\mathcal{S}', V_{21}^2 \setminus C)$ of the 3-Set Cover problem [7] is constructed. Note that the sizes of $\mathcal{S}'$ and $V_{21}^2 \setminus C$ are upper bounded by n as well. Afterwards, they called an approximate algorithm for the instance $I_{3\text{sc}} = (\mathcal{S}', V_{21}^2 \setminus C)$ of the 3-Set Cover problem, where the returned approximate solution is denoted by $\mathcal{S}_a$. It is easy to see that $\mathcal{S}_a \cup (\mathcal{S} \setminus \mathcal{S}')$ is a solution to $I_{\text{sc}} = (\mathcal{S}, V_{21}^2)$, and a depth restricted spanning tree of G can be obtained based on the solution and the construction way mentioned before. For the approximate ratio of the obtained spanning tree, it can be summarized in the following theorem.

Theorem 6.3. [1] *If there is an approximate algorithm for 3-Set Cover problem with ratio $r' > 1$, then there exists an approximate algorithm for DR-(1,2)-MSTP on G with ratio r, where r is given as follows.*

$$r = \begin{cases} 3r/4 & \text{if } r' > 5/3 \\ 5/4 & \text{otherwise} \end{cases}$$

6.1 Analysis of 3-Set Cover Problem

In the subsection, we focus on the 3-Set Cover problem and present a simple multi-objective evolutionary algorithm (named $\text{MOEA}_{3\text{sc}}$, given in Figure 5). The algorithm adopts the greedy strategy.

Assume that all sets in $\mathcal{S}'$ are numbered, i.e., $\mathcal{S}' = \{S_1, S_2, \ldots, S_{n'}\}$, where $n' = |\mathcal{S}'| \leq n$. The considered search space consists of all bit-strings with length n'. For a search point $x = x_1 \ldots x_{n'}$, the set S_i $(1 \leq i \leq n')$ is chosen iff $x_i = 1$. Denote by $S(x)$ the collection of sets that are chosen by x, by $V_c(x)$ the set of vertices that are covered by the sets in $S(x)$.

Consider a multi-objective fitness function $f_{3\text{sc}} = (f^1_{3\text{sc}}, \ldots, f^{n'}_{3\text{sc}}) : S \to \mathbb{N}^{n'}$ defined on the solution set. We first set $f_{3\text{sc}}(0^{n'}) = (0, \ldots, 0)$. For a solution y obtained by flipping exactly one 0-bit in x that is chosen uniformly at random and nothing else, if $x = 0^{n'}$, then we set

$$f_{3\text{sc}}(y) = (|V_c(y)|, 0, \ldots, 0);$$

otherwise,

$$f_{3\text{sc}}(y) = (f^1_{3\text{sc}}(x), \ldots, f^{|x|_1}_{3\text{sc}}(x), |V_c(y) \setminus V_c(x)|, 0, \ldots, 0).$$

Note that $f^i_{3\text{sc}}(y) = 0$ for all $|y|_1 + 1 \leq i \leq n'$.

Given two solutions x and y, x *strongly dominates* y with respect to $f_{3\text{sc}}$ if $|x|_1 = |y|_1$, and there exists an index $1 \leq j \leq n'$ such that $f^j_{3\text{sc}}(x) > f^j_{3\text{sc}}(y)$, and $f^i_{3\text{sc}}(x) = f^i_{3\text{sc}}(y)$ for all $1 \leq i \leq j-1$, written $x \succ_{3\text{sc}} y$. Solution x *dominates* y with respect to $f_{3\text{sc}}$ if $x \succ_{3\text{sc}} y$, or $|x|_1 = |y|_1$ and $f_{3\text{sc}}(x) = f_{3\text{sc}}(y)$, written $x \succcurlyeq_{3\text{sc}} y$. Thus two solutions are comparable with respect to $f_{3\text{sc}}$ only if they have the same Hamming weight.

The $\text{MOEA}_{3\text{sc}}$ starts with the population that is initialized with the solution $0^{n'}$. In each iteration, the algorithm chooses an individual x from the population, and generates an offspring by flipping exactly one 0-bit in x that is chosen uniformly at random and nothing else. If the offspring is not strongly dominated by any solution in the maintained population, then the offspring is included into the population, and all the other solutions that are dominated by the offspring (excluding itself) are discarded.

Theorem 6.4. *The expected time of the $\text{MOEA}_{3\text{sc}}$ to obtain an approximate solution with ratio $\frac{11}{6}$ to the instance $I_{3\text{sc}} = (\mathcal{S}', V^2_{21} \setminus C)$ of the 3-Set Cover problem is $O(n^3)$.*

Proof. We start with several related notions. Given a solution x, if $x = 0^{n'}$, or there is no solution that strongly dominates x with respect to $f_{3\text{sc}}$, then it is *potential*. Thus a potential solution can only be dominated by another potential solution with the same Hamming weight, with respect to $f_{3\text{sc}}$. Given a solution x, if $\sum_{i=1}^{|x|_1} f^i_{3\text{sc}}(x) = |V^2_{21} \setminus C|$, i.e., all vertices in $V^2_{21} \setminus C$ are covered, then it is a *complete solution* to the instance $I_{3\text{sc}}$ of the 3-Set Cover problem. Given a

Algorithm 5: $\text{MOEA}_{3\text{sc}}$

$S \leftarrow \{0^{n'}\}$;
while *stopping criterion not met* **do**
  Choose $x \in S$ uniformly at random;
  $y \leftarrow$ flip 0-bit x_i with $i \in [1, n']$ chosen uniformly at random;
  if $(0 \leq |y|_1 \leq n') \wedge (\nexists w \in S\colon w \succ_{3\text{sc}} y)$ **then**
    $S \leftarrow S \setminus \{z \in S \mid y \succcurlyeq_{3\text{sc}} z\}$;
    $S \leftarrow S \cup \{y\}$;

complete solution x, if $f^i_{3\text{sc}}(x) \neq 0$ for all $1 \leq i \leq |x|_1$, i.e., each set chosen by x makes a contribution to cover the vertices in $V^2_{21} \setminus C$, then it is a *minimal and complete solution* to $I_{3\text{sc}}$.

As the population maintained by the algorithm is initialized with $\{0^{n'}\}$, the population contains at least one potential solution. Now let x be the potential solution in the population with the maximum Hamming weight. In the following, we analyze the expected time of the algorithm to get a potential solution with Hamming weight exactly one larger than $|x|_1$.

Let S_i be the set of $\mathcal{S}' \setminus S(x)$ such that $|S_i \setminus V_c(x)|$ is maximized, i.e.,

$$S_i = \arg\max_{S \in \mathcal{S}' \setminus S(x)} |S \setminus V_c(x)|.$$

If the mutation chooses the potential solution x, and flips the 0-bit x_i corresponding to S_i and nothing else, then a potential solution x' with $|x'|_1 = |x|_1 + 1$ is obtained, which can be accepted by the algorithm as we assumed that x is the potential solution in the population with the maximum Hamming weight. The size of the population can be bounded by $n' + 1$, where $n' \leq n$, thus the mutation can be generated with probability $\Omega(1/n^2)$, i.e., the algorithm takes expected time $O(n^2)$ to get a potential solution with Hamming weight exactly one larger than $|x|_1$.

Summing over the waiting time for all possible Hamming weights of x, the $\text{MOEA}_{3\text{sc}}$ takes expected time $O(n^3)$ to get the potential solution x^* that is a minimal and complete solution to $I_{3\text{sc}}$.

By the definitions of potential solutions and minimal and complete solutions, we have that the solution x^* can be obtained by an execution of the greedy algorithm given in [6] for the 3-Set Cover problem. Thus the approximate ratio of x^* is $\frac{11}{6}$. □

Combining Theorems 6.3 and 6.4, we have the following theorem.

Theorem 6.5. *Using the $\text{MOEA}_{3\text{sc}}$ for the 3-Set Cover problem that can obtain an approximate solution with ratio $\frac{11}{6}$ to $I_{3\text{sc}} = (\mathcal{S}', V^2_{21} \setminus C)$ in expected time $O(n^3)$, an approximate solution with ratio $\frac{11}{8}$ for DR-(1,2)-MSTP on G can be constructed.*

Yu et al. [37] presented an evolutionary algorithm for the k-Set Cover problem that can obtain an approximate solution with ratio $H_k - \frac{k-1}{8k^9}$ in expected time $O(|C|^{k+1}|U|^2)$, where U denotes the universal set of instance $I = (C, U)$, C denotes the collection of the subsets of U, and $H_k = \sum_{i=1}^{k} \frac{1}{i}$ is the k-th harmonic number. That is, if $k = 3$, then the algorithm takes expected time $O(|C|^4|U|^2)$ to get a solution with approximate ratio $\frac{11}{6} - \frac{1}{4 \cdot 3^9}$. As the sizes of $\mathcal{S}'$

and $V_{21}^2 \setminus C$ in $I_{3sc} = (\mathcal{S}', V_{21}^2 \setminus C)$ can be upper bounded by n, we have the following theorem.

THEOREM 6.6. *Using the evolutionary algorithm for the 3-Set Cover problem given in [37] that can obtain an approximate solution with ratio* $\frac{11}{6} - \frac{1}{4 \cdot 3^9}$ *to* $I_{3sc} = (\mathcal{S}', V_{21}^2 \setminus C)$ *in expected time* $O(n^6)$*, an approximate solution with ratio* $\frac{11}{8} - \frac{1}{16 \cdot 3^8}$ *for DR-(1,2)-MSTP on G can be constructed.*

7 CONCLUSION

In the paper, we investigated a constrained version of the Minimum Spanning Tree problem, named Depth Restricted (1,2)-Minimum Spanning Tree problem (abbr. DR-(1,2)-MSTP), on a complete edge-weighted graph, in which each edge has weight 1 or 2. For the (1+1) EA, we compared its performance with respect to eight different fitness functions. With respect to the fitness functions f_2' and f_3', we showed that the (1+1) EA needs exponential expected time to get an improved solution if it starts with a given specific solution, but we did not consider the probability of the algorithm to get the specific solution starting with 0^m. Thus it is an interesting job to supplement the related results. With respect to the other six fitness functions, the (1+1) EA was shown to obtain an approximate solution with 2 or $\frac{3}{2}$ of DR-(1,2)-MSTP efficiently. Meanwhile, we gave a Multi-Objective Evolutionary Algorithm (abbr. MOEA) and showed that the MOEA can get a 2-approximate solution in polynomial expected time. As the large population size of the MOEA slows its optimization process, we considered it two variants, where any of them maintains a population with size at most two. The analysis for the performance of the two variants showed that the interplay between the two solutions in the population (a feasible one and an infeasible one) can promote the accelerate the optimization process.

Additionally, by introducing several local search operations, we studied the performance of the four algorithms mentioned above to get an improved approximate solution with ratio $\frac{3}{2}$. Finally, we reformulated DR-(1,2)-MSTP in form of the Set Cover problem following Alfandari and Paschos [1], and investigated it from the perspective of the Set Cover problem, leading to an approximate ratio better than $\frac{3}{2}$.

It is not hard to see that the reasoning given in the paper can be adapted to DR-(1,α)-MSTP on a complete edge-weighted graph, in which each edge has weight 1 or α (α is an integer not less than 2). Future work on evolutionary algorithms for the *generalized* DR-MSTP would be very interesting, where the depth restriction is relaxed to an integer $\beta > 2$, or the weight of each edge in the input graph has more than two options.

ACKNOWLEDGMENTS

The research leading to these results has received funding from National Natural Science Foundation of China under Grants 61802441, 61672536, 61836016, and the Australian Research Council under Grant DP160102401.

REFERENCES

[1] Laurent Alfandari and Vangelis Th Paschos. 1999. Approximating minimum spanning tree of depth 2. *International Transactions in Operational Research* 6, 6 (1999), 607–622.

[2] Eric Angel, Evripidis Bampis, and Laurent Gourvés. 2004. Approximating the Pareto curve with local search for the bicriteria TSP (1, 2) problem. *Theoretical Computer Science* 310, 1-3 (2004), 135–146.

[3] AE Baburin, Federico Della Croce, E Kh Gimadi, Yu V Glazkov, and V Th Paschos. 2009. Approximation algorithms for the 2-peripatetic salesman problem with edge weights 1 and 2. *Discrete Applied Mathematics* 157, 9 (2009), 1988–1992.

[4] Piotr Berman and Marek Karpinski. 2006. 8/7-approximation algorithm for (1, 2)-TSP. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*. Society for Industrial and Applied Mathematics, 641–648.

[5] Marshall Bern and Paul Plassmann. 1989. The Steiner problem with edge lengths 1 and 2. *Inform. Process. Lett.* 32, 4 (1989), 171–176.

[6] Vasek Chvatal. 1979. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research* 4, 3 (1979), 233–235.

[7] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2009. *Introduction to algorithms*. MIT press.

[8] Dogan Corus, Per Kristian Lehre, and Frank Neumann. 2013. The generalized minimum spanning tree problem: a parameterized complexity analysis of bi-level optimisation. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 519–526.

[9] Marek Cygan, Fedor V Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. 2015. *Parameterized algorithms*. Vol. 4. Springer.

[10] Benjamin Doerr, Daniel Johannsen, and Carola Winzen. 2012. Multiplicative drift analysis. *Algorithmica* 64 (2012), 673–697.

[11] Rodney G Downey and Michael Ralph Fellows. 2012. *Parameterized complexity*. Springer Science & Business Media.

[12] Tobias Friedrich, Jun He, Nils Hebbinghaus, Frank Neumann, and Carsten Witt. 2010. Approximating covering problems by randomized search heuristics using multi-objective models. *Evolutionary Computation* 18, 4 (2010), 617–633.

[13] Wanru Gao, Mojgan Pourhassan, Vahid Roostapour, and Frank Neumann. 2019. Runtime analysis of evolutionary multi-objective algorithms optimising the degree and diameter of spanning trees. In *Proceedings of the Conference on Evolutionary Multi-Criterion Optimization*. Springer, 504–515.

[14] Martin Gruber, Jano van Hemert, and Günther R Raidl. 2006. Neighbourhood searches for the bounded diameter minimum spanning tree problem embedded in a VNS, EA, and ACO. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 1187–1194.

[15] Jun He and Xin Yao. 2004. A study of drift analysis for estimating computation time of evolutionary algorithms. *Natural Computing* 3 (2004), 21–35.

[16] Vojtěch Jarník. 1930. O jistém problému minimálním. *Práca Moravské Prírodovedecké Spolecnosti* 6 (1930), 57–63.

[17] Stefan Kratsch, Per Kristian Lehre, Frank Neumann, and Pietro Simone Oliveto. 2010. Fixed parameter evolutionary algorithms and maximum leaf spanning trees: A matter of mutation. In *Proceedings of the Conference on Parallel Problem Solving from Nature*. Springer, 204–213.

[18] Stefan Kratsch and Frank Neumann. 2013. Fixed-parameter evolutionary algorithms and the vertex cover problem. *Algorithmica* 65, 4 (2013), 754–771.

[19] Joseph B Kruskal. 1956. On the shortest spanning subtree of a graph and the traveling salesman problem. *Proc. Amer. Math. Soc.* 7, 1 (1956), 48–50.

[20] Samadhi Nallaperuma, Frank Neumann, and Dirk Sudholt. 2017. Expected fitness gains of randomized search heuristics for the traveling salesperson problem. *Evolutionary Computation* 25, 4 (2017), 673–705.

[21] Frank Neumann. 2007. Expected runtimes of a simple evolutionary algorithm for the multi-objective minimum spanning tree problem. *European Journal of Operational Research* 181, 3 (2007), 1620–1629.

[22] Frank Neumann and Andrew M Sutton. 2018. Runtime analysis of evolutionary algorithms for the knapsack problem with favorably correlated weights. In *Proceedings of the Conference on Parallel Problem Solving from Nature*. Springer, 141–152.

[23] Frank Neumann and Ingo Wegener. 2006. Minimum spanning trees made easier via multi-objective optimization. *Natural Computing* 5 (2006), 305–319.

[24] Frank Neumann and Ingo Wegener. 2007. Randomized local search, evolutionary algorithms, and the minimum spanning tree problem. *Theoretical Computer Science* 378, 1 (2007), 32–40.

[25] Frank Neumann and Carsten Witt. 2010. Ant colony optimization and the minimum spanning tree problem. *Theoretical Computer Science* 411, 25 (2010), 2406–2413.

[26] Frank Neumann and Carsten Witt. 2015. On the runtime of randomized local search and simple evolutionary algorithms for dynamic makespan scheduling. In *Proceedings of the International Joint Conference on Artificial Intelligence*. 3742–3748.

[27] Christos H Papadimitriou and Mihalis Yannakakis. 1993. The traveling salesman problem with distances one and two. *Mathematics of Operations Research* 18, 1 (1993), 1–11.

[28] Mojgan Pourhassan, Tobias Friedrich, and Frank Neumann. 2017. On the use of the dual formulation for minimum weighted vertex cover in evolutionary algorithms. In *Proceedings of the Workshop on Foundations of Genetic Algorithms*. ACM, 37–44.

[29] Mojgan Pourhassan, Feng Shi, and Frank Neumann. 2016. Parameterized analysis of multi-objective evolutionary algorithms and the weighted vertex cover problem. In *Proceedings of the Conference on Parallel Problem Solving from Nature*. Springer, 729–739.

[30] Günther R Raidl and Bryant A Julstrom. 2003. Greedy heuristics and an evolutionary algorithm for the bounded-diameter minimum spanning tree problem. In *Proceedings of the ACM Symposium on Applied Computing*. ACM, 747–752.

[31] Vahid Roostapour, Aneta Neumann, and Frank Neumann. 2018. On the performance of baseline evolutionary algorithms on the dynamic knapsack problem. In *Proceedings of the Conference on Parallel Problem Solving from Nature*. Springer, 158–169.

[32] Feng Shi, Frank Neumann, and Jianxin Wang. 2018. Runtime analysis of randomized search heuristics for the dynamic weighted vertex cover problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 1515–1522.

[33] Feng Shi, Martin Schirneck, Tobias Friedrich, Timo Kötzing, and Frank Neumann. 2019. Reoptimization time analysis of evolutionary algorithms on linear functions under dynamic uniform constraints. *Algorithmica* 81, 2 (2019), 828–857.

[34] Andrew M Sutton and Frank Neumann. 2012. A parameterized runtime analysis of simple evolutionary algorithms for makespan scheduling. In *Proceedings of the Conference on Parallel Problem Solving from Nature*. Springer, 52–61.

[35] Andrew M Sutton, Frank Neumann, and Samadhi Nallaperuma. 2014. Parameterized runtime analyses of evolutionary algorithms for the planar euclidean traveling salesperson problem. *Evolutionary Computation* 22, 4 (2014), 595–628.

[36] Junhua Wu, Sergey Polyakovskiy, and Frank Neumann. 2016. On the impact of the renting rate for the unconstrained nonlinear knapsack problem. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, 413–419.

[37] Yang Yu, Xin Yao, and Zhi-Hua Zhou. 2012. On the approximation ability of evolutionary optimization with application to minimum set cover. *Artificial Intelligence* 180 (2012), 20–33.