Networks

WILEY

RESEARCH ARTICLE OPEN ACCESS

Tractable but Hard to Approximate: The Bi-Objective Minimum *s*-*t*-Cut Problem With Binary Capacities

Jan Boeckmann[1] | Stephan Helfrich[2] | Oliver Bachtler[3] | Stefan Ruzika[3] | Clemens Thielen[4]

[1]Johannes Kepler University Linz, Institute for Business Analytics and Technology Transformation, Linz, Austria | [2]Karlsruhe Institute of Technology, Operations of Critical Infrastructures, Institute of Information Security and Dependability (KASTEL), Karlsruhe, Germany | [3]RPTU Kaiserslautern-Landau, Department of Mathematics, Kaiserslautern, Germany | [4]Technical University of Munich, Campus Straubing for Biotechnology and Sustainability, Straubing, Germany

Correspondence: Jan Boeckmann (jan.boeckmann@jku.at)

Received: 23 April 2025 | **Revised:** 27 November 2025 | **Accepted:** 6 December 2025

Keywords: approximation | inapproximability | minimum *s*-*t*-cut problem | multi-objective optimization

ABSTRACT

The minimum *s*-*t*-cut problem is one of the most-studied problems in discrete optimization and has a unique complexity status in multi-objective optimization. Even though the single-objective version of the problem can be solved in polynomial time, it has been shown in the seminal work of Papadimitriou and Yannakakis (2000) that there does not exist a multi-objective fully polynomial-time approximation scheme (MFPTAS) for the minimum *s*-*t*-cut problem unless $\mathcal{P} = \mathcal{NP}$. This holds both for the case of $p \geq 3$ objective functions with arc capacities in $\{0, 1\}$ and for $p = 2$ objective functions with general capacities, and even for tractable instances where the number of non-dominated points is only quadratic in the input size. In this article, we strengthen these results by showing that, assuming $\mathcal{P} \neq \mathcal{NP}$, there does not exist an MFPTAS for the minimum *s*-*t*-cut problem with two objectives and arc capacities in $\{(1, 0), (0, 1)\}$, nor for the minimum *s*-*t*-cut problem with two objectives and arc capacities in $\{(0, 1), (1, 1)\}$. This advancement is particularly interesting since the considered problem variants are the only known problems in multi-objective optimization that do not admit an MFPTAS even though their single-objective versions are solvable in polynomial time and the problems are *tractable*, that is, the numbers of non-dominated points are polynomial (even linear) in the input size. Furthermore, we complement this result by showing that, on graphs of bounded tree-width, the minimum *s*-*t*-cut problem with polynomially bounded arc capacities can be solved exactly in polynomial time for any constant number of objectives.

1 | Introduction

The minimum *s*-*t*-cut problem is one of the most important problems in graph theory and discrete optimization. Given two distinct nodes *s* and *t* in a directed graph with capacities on the arcs, it asks for an *s*-*t*-cut with minimum capacity. The single-objective version of the problem can be solved efficiently in polynomial time and is applicable to a wide range of real-world problems [1]. Moreover, algorithms for the minimum *s*-*t*-cut problem are used as subroutines in many discrete optimization algorithms [2–5].

In this article, we consider the multi-objective version of the minimum *s*-*t*-cut problem, which—as outlined in the following—has a unique complexity status in multi-objective optimization.

In a multi-objective optimization problem, where multiple, often conflicting objective functions are considered, solutions that optimize all objective functions simultaneously usually do not exist. Instead, one is interested in so-called *efficient solutions* for which any solution that is better in one objective is necessarily

worse in one of the other objectives. The image of an efficient solution under the vector-valued objective function is called a *non-dominated point*. The aim in a multi-objective optimization problem then consists of computing the whole set of non-dominated points and, for each of them, at least one corresponding efficient solution [6].

Unfortunately, multi-objective optimization problems are often very hard to solve since (1) the number of non-dominated points is usually super-polynomial in the input for discrete problems (and typically infinite for continuous problems), that is, the problem is *intractable*, and (2) it is often $\mathcal{NP}$-hard to decide whether a given solution is efficient or whether a given vector of objective values is non-dominated (see [7, 8]). Both issues are a strong motivation to study *approximations* of multi-objective optimization problems. Generalizing the well-known (multiplicative) notion of approximation from the single-objective case, an *α-approximation* for $\alpha \geq 1$ is a set of solutions such that, for each feasible solution, there exists a solution in the set that approximates it with an approximation factor of α in each objective function. An *α-approximation algorithm* is an algorithm that computes an α-approximation in polynomial time for every instance, and a *multi-objective fully polynomial-time approximation scheme (MFPTAS)* is a family of algorithms that contains a $(1+\varepsilon)$-approximation for every $\varepsilon > 0$ such that the running time is not only polynomial in the input size, but also in $1/\varepsilon$. Approximation algorithms for multi-objective optimization problems have been studied intensively since the early 2000s and represent an active area of research. For a recent survey, we refer to [9].

While most discrete multi-objective optimization problems are intractable, $(1+\varepsilon)$-approximations exist for every $\varepsilon > 0$ under weak assumptions as shown in the seminal work of Papadimitriou and Yannakakis [10], and an MFPTAS can be derived for many important problems by either problem-specific algorithms or general approximation methods [9]. General approximation methods that can be applied to the minimum s-t-cut problem studied here include the methods from [11, 12], which yield $(p+\varepsilon)$-approximation algorithms for the p-objective problem, and the method from [13], which yields approximation factors of $1+2\varepsilon$ and $1+\frac{2}{\varepsilon}$ in the first and the second objective, respectively, for every $0 < \varepsilon \leq 1$ in the bi-objective case.

Inapproximability results for multi-objective optimization problems, however, often derive from known inapproximability results for the single-objective version of the problem. The multi-objective minimum s-t-cut problem is unique in this respect since no MFPTAS can exist for this problem unless $\mathcal{P} = \mathcal{NP}$ [10] even though the single-objective version is polynomially solvable—and this holds even for the tractable case of three-objective functions and arc capacities in $\{0,1\}$, where the number of non-dominated points is only quadratic in the input size (and also for bi-objective problem instances with general capacities derived from such three-objective instances). This unique status provides a strong motivation for further investigation of the complexity and approximability of the multi-objective minimum s-t-cut problem.

From a different point of view, one can interpret the multi-objective minimum s-t-cut problem as a (single-objective) budgeted minimum s-t-cut problem, where one objective represents the capacities on the arcs and upper bounds on the other objectives are imposed to represent budget constraints on the cut. The budgeted minimum s-t-cut problem is known to be $\mathcal{NP}$-hard even in the case of only one budget constraint [14] and is an active area of research [15]. Further, it has been shown to be equivalent to the well-known network flow interdiction problem, which is known to be hard to approximate [2].

1.1 | Our Contribution

We strengthen the result from [10] and show that, unless $\mathcal{P} = \mathcal{NP}$, there does not exist an MFPTAS for the multi-objective minimum s-t-cut problem with only *two* objective functions and arc capacities in $\{(1,0),(0,1)\}$ or $\{(0,1),(1,1)\}$. To this end, we first provide a general criterion for the existence of an MFPTAS. Next, we extend and adapt two reductions from the literature on the budgeted minimum s-t-cut problem [14, 16] to our multi-objective problem formulation and apply the previously-derived criterion to prove the non-existence of an MFPTAS for the *bi-objective* minimum s-t-cut problem with arc capacities in $\{(1,0),(0,1)\}$, and for the bi-objective minimum s-t-cut problem with arc capacities in $\{(0,1),(1,1)\}$. By doing so, we slightly strengthen the results of [14, 16] and obtain that the budgeted minium s-t-cut problem is strongly $\mathcal{NP}$-hard even for one budget constraint, binary capacities and costs on the arcs such that, for each arc, exactly one of these two values equals one, and even for one budget constraint, binary arc capacities, and a constant arc cost of one for each arc.

Since the cardinality of the non-dominated set in the bi-objective problem is at most $m+1$ (where m denotes the number of arcs in the graph), our results show that even bi-objective problems for which the number of non-dominated points is *linear* in the input size do not necessarily admit an MFPTAS even if their single-objective version is polynomially solvable. To complement this result, we show that the multi-objective minimum s-t-cut problem with polynomially bounded arc capacities can be solved in polynomial time for any constant number of objectives on graphs of bounded tree-width. This result yields the first polynomial-time algorithm for the budgeted minimum s-t-cut problem on graphs with bounded tree-width if the capacities and costs are polynomially bounded.

2 | Preliminaries

An *instance* of a multi-objective optimization problem consists of a set X of feasible solutions, a vector-valued objective function $f : X \to \mathbb{R}^p_{\geq 0}$ [1] and an *optimization sense* max or min. The ith component of f is also referred to as the ith *objective function* and denoted by f_i. A *multi-objective optimization problem* is given by the set of its instances. The image $Y := f(X)$ of X under f is called the *image set* and each element of Y is called a *feasible point*.

Since solutions that optimize all objectives simultaneously do usually not exist in multi-objective optimization problems, the solutions of interest are those that cannot be improved upon with respect to any objective function without worsening the value of

at least one other objective. These *efficient solutions* are formally defined as follows.

Definition 1. A solution $x \in X$ *dominates* another solution $x' \in X$ if x is at least as good as x' in each objective and strictly better in at least one objective. A solution that is not dominated by any other solution is called *efficient*. The set X_E of all efficient solutions is called the efficient set. The image $f(x) \in Y$ of an efficient solution $x \in X$ is called a non-dominated point, and the set Y_N of all non-dominated points is called the non-dominated set.

As discussed above, multi-objective problems are often hard to solve, which motivates to consider approximations and approximation algorithms.

Definition 2. Let $\alpha \geq 1$. A feasible solution $x \in X$ *α-approximates* another feasible solution $x' \in X$ (or, equivalently, the feasible point $y = f(x)$ *α-approximates* the feasible point $y' = f(x')$) if

$f_i(x) \geq 1/\alpha \cdot f_i(x') \forall i \in \{1, \ldots, p\}$ if the optimization sense is max, or

$f_i(x) \leq \alpha \cdot f_i(x') \forall i \in \{1, \ldots, p\}$ if the optimization sense is min.

A set $X' \subseteq X$ of feasible solutions is called an *α-approximation* if, for each feasible solution $x \in X$, there exists a solution $x' \in X'$ that α-approximates x.

Definition 3. An *α-approximation algorithm* for a multi-objective optimization problem P is an algorithm ALG that, for each instance of the problem, computes an α-approximation in time polynomial in the encoding size of the instance.

Further, a multi-objective polynomial-time approximation scheme (MPTAS) for P is a family $(\mathrm{ALG}_\varepsilon)_{\varepsilon>0}$ of algorithms such that, for each $\varepsilon > 0$, the algorithm ALG_ε is a $(1+\varepsilon)$-approximation algorithm. An MPTAS $(\mathrm{ALG}_\varepsilon)_{\varepsilon>0}$ for P is called a multi-objective fully polynomial-time approximation scheme (MFPTAS) if the running time of each algorithm ALG_ε is additionally polynomial in $1/\varepsilon$.

Note that, for $\alpha = 1$, an α-approximation is equivalent to a set of feasible solutions whose image contains the whole non-dominated set, so a 1-approximation algorithm for a multi-objective optimization problem is a (polynomial-time) exact algorithm.

3 | A Criterion for the Existence of an MFPTAS

We start by characterizing the existence of an MFPTAS by the existence of an exact polynomial-time algorithm for the considered multi-objective problem.

In this section, we consider multi-objective optimization problems whose possible objective values are natural numbers that are polynomially bounded. Formally, we make the following assumption:

Assumption 4. We assume that $Y \subseteq \mathbb{N}_0^p$. Further, we assume that $\mathrm{ub} := \max_{\substack{x \in X \\ i \in \{1,\ldots,p\}}} f_i(x) + 1$ is polynomial in the input size.

Note that, even in the single-objective case, the restriction to integer instead of rational objective values can make the difference between weak and strong $\mathcal{NP}$-hardness [17] or represent a prerequisite for important general results concerning the existence of approximation schemes [18].

The following theorem formalizes and generalizes an idea that is sketched for the case of the minimum s-t-cut problem in the proof of Theorem 6 in [10].

Theorem 5. *Let P be a multi-objective optimization problem that satisfies Assumption 4. Then there exists an MFPTAS for P if and only if there exists a polynomial-time exact algorithm for P.*

Proof. We only provide the proof for the optimization sense min. The proof for optimization sense max is along the same lines.

For the forward direction, assume that there exists an MFPTAS for P. We set $\varepsilon := 1/\mathrm{ub}$ and obtain a set $X_\varepsilon \subseteq X$ in polynomial time by applying the MFPTAS for ε. Since all objective values are integer, it holds that $|f_i(x) - f_i(x')| \in \mathbb{N}_0$ for all $x,\ x' \in X$. Now let $y \in Y_N$ be any non-dominated point. Since the set X_ε is obtained from an MFPTAS, there exists a solution $x' \in X_\varepsilon$ with $f_i(x') \leq (1+\varepsilon) \cdot y_i$ for all $i \in \{1, \ldots, p\}$. Rearranging the inequality and plugging in $\varepsilon = 1/\mathrm{ub}$ yields $f_i(x') - y_i \leq y_i/\mathrm{ub} < 1$. Since all objective values are natural numbers, this means that $f_i(x') - y_i \leq 0$ and, since y is non-dominated, it follows that $f_i(x') - y_i = 0$ for all $i \in \{1, \ldots, p\}$, that is, $f(x) = y$. This shows that $f(X_\varepsilon) \supseteq Y_N$, that is, applying the MFPTAS for $\varepsilon = 1/\mathrm{ub}$ yields an exact algorithm for P. Its running time is polynomial by Assumption 4.

The inverse direction is clear, since any exact polynomial-time algorithm is also an MFPTAS. □

As a final remark for this section, we note that the result of Theorem 5 also holds for the single-objective case. This means that, if all objective values of a single-objective problem are natural numbers and the maximum possible objective value is polynomial in the encoding length of the input instance, there exists an exact polynomial-time algorithm if and only if there exists an FPTAS (which is the equivalent to an MFPTAS in the single-objective case).

4 | The Multi-Objective Minimum s-t-Cut Problem

In this section, we formally introduce the multi-objective minimum s-t-cut problem. To this end, we start by introducing some definitions and notation.

A *(directed) p-capacity network* $G = (V, R, u)$ is a triple consisting of a *vertex set* V, an *arc set* R, and a *capacity function* $u : R \to \mathbb{R}^p_{\geq 0}$. The *capacity* of a set $\overline{R} \subseteq R$ of arcs is defined as $u(\overline{R}) := \sum_{r \in \overline{R}} u(r)$. We denote the start vertex of an arc $r \in R$ by $\alpha(r)$ and the end vertex by $\omega(r)$. Further, for sets $U,\ U' \subseteq V$, we define $R(U, U') := \{r \in R : \alpha(r) \in U \text{ and } \omega(r) \in U'\}$. In the case that U or U' is a singleton set, that is, it is just one vertex, we drop the braces for the ease of notation.

For two vertices $s \neq t$, called the *source* and the *terminal*, respectively, an *s-t-cut* $C = (S, T)$, also called *cut* in the following, is a

partition $V = S \dot{\cup} T$ of the vertices of G such that $s \in S$ and $t \in T$, and we say that the arcs in $R(S,T)$ are *in the cut* C. For an arc r being in a cut C, we slightly abuse notation and write $r \in C$. The *capacity* of a cut $C = (S,T)$ in G is defined as $\text{cap}^G(C) := \sum_{r \in C} u(r)$.

Given a p-capacity network and two vertices $s \neq t$ in the network, the aim in the *multi-objective minimum $s-t$-cut problem* (MMCP) is to find a set $\mathcal{C}$ of cuts such that its image is the non-dominated set Y_N.

Throughout the article, we outline connections between our results and the budgeted minimum s-t-cut problem, which can be interpreted as the budget-constrained version of the multi-objective problem where upper bounds are enforced on all except the first objective. In this single-objective problem, each arc has only one capacity value, but for each arc r, there is a set of k additional costs $b^{(i)}(r)$, $i = 1, \ldots, k$. Then, given k budgets $B(i)$, $i = 1, \ldots, k$, the budgeted minimum s-t-cut problem asks for an s-t-cut C of minimum capacity subject to the budget constraints $\sum_{r \in C} b^{(i)}(r) \leq B(i)$ for $i = 1, \ldots, k$.

In the next section, we show that, unless $\mathcal{P} = \mathcal{NP}$, there does not exist an MFPTAS for MMCP with $p = 2$ and capacities $u(r)$ in $\{(1,0),(0,1)\}$ or $\{(0,1),(1,1)\}$ for all $r \in R$.

5 | Hardness of Approximation

In this section, we show that MMCP is $\mathcal{NP}$-hard even for two objectives where the capacities are restricted to $\{(1,0),(0,1)\}$ or $\{(0,1),(1,1)\}$. From a holistic viewpoint, this implies that restricting the capacities in any meaningful way (which does not make one of the objectives obsolete) leaves little hope to make the problem polynomially solvable.

5.1 | Hardness of Approximation for Capacities in $\{(1,0),(0,1)\}$

We first show that MMCP is $\mathcal{NP}$-hard even for two objectives and when each arc capacity is required to be one of the two unit vectors in $\mathbb{R}^2$. To this end, we slightly strengthen the reduction idea of [16], who show that the budgeted minimum s-t-cut problem with one budget constraint is strongly $\mathcal{NP}$-hard, and adapt it to our problem definition. In particular, our reduction does not require arcs with infinite cost, which are needed in the reduction presented in [16]. As a consequence, our result implies strong $\mathcal{NP}$-hardness even for one budget constraint and binary capacities and costs on the arcs such that, for each arc, exactly one of these two values equals one (Corollary 8).

The reduction will be from the minimum vertex cover problem on bipartite graphs (MVCPBG), which is formally defined as follows:

Minimum vertex cover problem on bipartite graphs (MVCPBG)	
INSTANCE:	An undirected, bipartite graph $H = (V_l \cup V_r, E)$, integers B_l, B_r.
TASK:	Is there a vertex cover $X \subseteq V_l \cup V_r$ with $\lvert X \cap V_l \rvert \leq B_l$ and $\lvert X \cap V_r \rvert \leq B_r$?

A proof of strong $\mathcal{NP}$-hardness of MVCPBG can be found in [19].

For the reduction, consider an instance of MVCPBG given by a bipartite graph $H = (V_l \cup V_r, E)$ and two integers B_l, B_r. We construct an instance of BMCPBC consisting of a 2-capacity network $G = (V, R, u)$ as follows. For the vertex set, we take the same set as in H and add the source s and the terminal t, that is, $V = V_l \cup V_r \cup \{s,t\}$. To construct the arc set R, we set $n_l := |V_l|$ and $n_r := |V_r|$. For each vertex $v \in V_l$, we introduce an arc from s to v with capacity $(1,0)$. For each vertex $v \in V_r$, we introduce an arc from v to t with capacity $(0,1)$. For each edge $e \in E$ connecting vertices $v_l \in V_l$ and $v_r \in V_r$, we add $n_l + 1$ parallel arcs with capacity $(1,0)$ and $n_r + 1$ parallel arcs with capacity $(0,1)$ from v_l to v_r. The resulting graph is illustrated in Figure 1.

To complete the reduction, we show that there exists a vertex cover $X \subseteq V_l \cup V_r$ with $|X \cap V_l| \leq B_l$ and $|X \cap V_r| \leq B_r$ in the given instance of MVCPBG if and only if there exists an efficient minimum cut in the constructed instance of MMCP with capacity less than or equal to (B_l, B_r).

First, let $X \subseteq V_l \cup V_r$ be a vertex cover with $|X \cap V_l| \leq B_l$ and $|X \cap V_r| \leq B_r$. We consider the cut (S,T) in G given by $S := \{s\} \cup (V_l \setminus X) \cup (V_r \cap X)$ and $T := \{t\} \cup (V_l \cap X) \cup (V_r \setminus X)$. This cut then has capacity $(|X \cap V_l|, |X \cap V_r|)$. Hence, there exists an efficient cut with capacity less than or equal to (B_l, B_r).

Conversely, let (S,T) be an efficient cut of capacity (k_l, k_r) with $k_l \leq B_l$ and $k_r \leq B_r$. We construct a vertex cover X of H as follows: For a vertex $v \in V_l$, we include it in X if and only if $v \in T$. For a vertex $v \in V_r$, we include it in X if and only if $v \in S$. Since the lexicographic solutions have capacities $(n_l, 0)$ and $(0, n_r)$, we know that, for an edge $(v_l, v_r) \in E$, it cannot hold that $v_l \in S$ and $v_r \in T$. If this were the case, the capacity of the cut (S,T) would be at least $(n_l + 1, n_r + 1)$, contradicting its efficiency. This means that, for each edge $(v_l, v_r) \in E$, it holds that $v_l \in T$ or $v_r \in S$. By construction, this means that $v_l \in X$ or $v_r \in X$, which proves that X is indeed a vertex cover with $k_l \leq B_l$ vertices in V_l and $k_r \leq B_r$

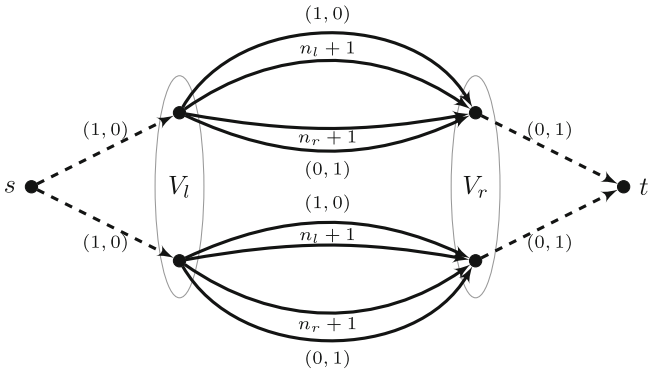

FIGURE 1 | An illustration of the 2-capacity network G constructed in the reduction from MVCPBG to MMCP with capacities in $\{(1,0),(0,1)\}$. Individual arcs are dashed, while a bundle of parallels is depicted by two solid arcs with the number of arcs in the respective bundle between them. The figure depicts two vertices in V_l and two in V_r that are connected by two edges in the graph H. Vertices that are not connected in H are also not connected in G.

vertices in V_r. This completes the reduction and proves the following theorem.

Theorem 6. *MMCP is strongly $\mathcal{NP}$-hard even for two objectives and capacities in* $\{(1,0),(0,1)\}$.

Our second main result is an immediate implication of Theorem 6 and strengthens its statement.

Theorem 7. *Unless $\mathcal{P} = \mathcal{NP}$, there does not exist an MFPTAS for MMCP, even for two objectives and capacities in* $\{(1,0),(0,1)\}$.

Proof. Since MMCP with the assumptions from the theorem satisfies Assumption 4, the claim follows directly from Theorems 5 and 6. □

Further, as already outlined in the introduction of this section, we obtain a slightly stronger hardness result for the budgeted minimum s-t-cut problem:

Corollary 8. *The budgeted minimum s-t-cut problem with one budget constraint and binary capacities and costs on the arcs such that, for each arc, exactly one of these two values equals one, is strongly$\mathcal{NP}$-hard. In particular, the problem does not admit an FPTAS unless$\mathcal{P} = \mathcal{NP}$.*

5.2 | Hardness of Approximation for Capacities in $\{(0,1),(1,1)\}$

In this section, we provide a reduction that shows that MMCP is $\mathcal{NP}$-hard even for two objectives and capacities in $\{(0,1),(1,1)\}$. In particular, the second objective is constant in this setting. The reduction follows a similar idea as the reduction presented in [14], who show that the budgeted minium s-t-cut problem with constant arc costs of one is strongly $\mathcal{NP}$-hard. The reduction in [14], however, uses general arc capacities that are not restricted to $\{0,1\}$. Thus, the result in [14] does not yield a stronger result for MMCP than the one presented in [10].

The reduction will be from BISECTION, which is formally defined as follows.

BISECTION	
INSTANCE:	An undirected graph $H = (V', E)$ with $2n$ vertices and m edges and an integer B.
TASK:	Is there partition of the vertices in V' into V'_1 and V'_2 with $\lvert V'_1\rvert = \lvert V'_2\rvert = n$ such that the number of edges that have one endpoint in V'_1 and one endpoint in V'_2 is less than or equal to B?

A proof of strong $\mathcal{NP}$-hardness of BISECTION can be found in [20].

Given an instance of BISECTION, we construct an instance of MMCP, that is, a 2-capacity network $G = (V, R, u)$ as follows. We start by setting $V := V'$ and, for each edge $e \in E$, we add an arc r from one endpoint of e to the other with capacity $u(r) = (1,1)$ and its inverse arc with the same capacity. Next, we add a vertex s to V and, for each $v \in V'$, we add $n+m$ parallel arcs with capacity $(1,1)$ from s to v. Finally, we add another vertex t to V and add one arc from each vertex in V' to t with capacity $(1,1)$ and another $n(n+m)+m-1$ parallel arcs with capacity $(0,1)$. This is illustrated in Figure 2.

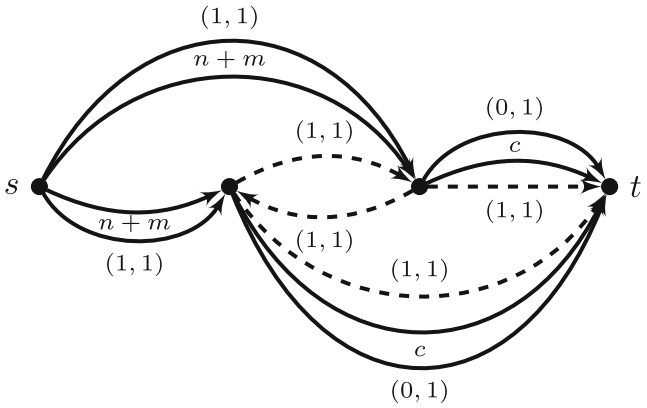

FIGURE 2 | An illustration of the 2-capacity network constructed in the reduction from BISECTION with $n = 1$ and one edge to MMCP with capacities in $\{(0,1),(1,1)\}$. Individual arcs are dashed, while a bundle of parallels is depicted by two solid arcs with the number of arcs in the respective bundle between them. The figure depicts two vertices in V' that are connected by an edge in the graph H, where the numbers between the sets of parallel arcs denote how many copies are in the respective bundle and $c := n(n+m)+m-1$. Vertices that are not connected in H are also not connected in G.

We claim that there exists a bisection of $H = (V', E)$ with at most B edges if and only if there exists a cut in the constructed instance with capacity at most $(n(n+m+1)+B, n(n+1)(n+m)+nm+B)$.

We start with the forward direction. To this end, let V_1 and V_2 be a bisection of V' containing at most B edges. We set $S := \{s\} \cup V_1$ and $T := \{t\} \cup V_2$, which implies that $S \cap V' = V_1$ and $T \cap V' = V_2$. The resulting cut contains $n \cdot (n+m)$ arcs of capacity $(1,1)$ in $R(s, V_2)$, n arcs with capacity $(1,1)$, and $n \cdot (n(n+m)+m)$ arcs with capacity $(0,1)$ in $R(V_1, t)$, and at most B arcs with capacity $(1,1)$ in $R(V_1, V_2)$. Thus, the capacity of the cut is at most $(n(n+m+1)+B, n(n+1)(n+m)+nm+B)$ as required.

Conversely, let $C = (S, T)$ be a cut of capacity at most $(n(n+m+1)+B, n(n+1)(n+m)+nm+B)$. We set $V_1 := V' \cap S$ and $V_2 := V' \cap T$ and show that $\lvert V_1\rvert = \lvert V_2\rvert = n$. We note that $\lvert V_1\rvert \leq n$ since, otherwise, there are at least $(n+1) \cdot (n(n+m)+m)$ arcs in $R(V_1, t)$, each of which contributes a capacity of 1 to the second objective. Since we may assume that $B < m$, we obtain a contradiction to the second objective of the cut attaining a value strictly larger than $n(n+1)(n+m)+nm+B$.

Furthermore, $\lvert V_1\rvert \geq n$ since, otherwise, there are at least $(n+1) \cdot (n+m)$ arcs of capacity $(1,1)$ in $R(s, V_2)$. Since we may, again, assume that $B < m$, we obtain a contradiction to the first objective of the cut attaining a value strictly larger than $n(n+m+1)+B$. Since $\lvert V_1\rvert + \lvert V_2\rvert = 2n$, this implies that $\lvert V_1\rvert = \lvert V_2\rvert = n$.

To complete the reduction, it remains to show that the number of edges between V_1 and V_2 is at most B. Since $|V_1| = |V_2| = n$, the cut (S, T) contains $n \cdot (n+m)$ arcs in $R(s, V_2)$ and $n \cdot (n(n+m)+m)$ arcs in $R(V_1, t)$, for a combined total of $(n+1)n(n+m)+nm$. Since the second capacity value is always 1, this leaves at most B arcs in $R(V_1, V_2)$ and, thus, between V_1 and V_2, which completes the reduction. This yields the following theorems.

Theorem 9. *MMCP is strongly $\mathcal{NP}$-hard even for two objectives and capacities in $\{(0,1),(1,1)\}$.*

Theorem 10. *Unless $\mathcal{P} = \mathcal{NP}$, there does not exist an MFPTAS for MMCP, even for two objectives and capacities in $\{(0,1),(1,1)\}$.*

Our result strengthens the result of [14] and we obtain that the budgeted minimum s-t-cut problem with constant arc costs of one remains strongly $\mathcal{NP}$-hard even when arc capacities are restricted to be binary:

Corollary 11. *The budgeted minimum s-t-cut problem with one budget constraint, binary arc capacities, and a constant arc cost of one for each arc is strongly $\mathcal{NP}$-hard. In particular, the problem does not admit an FPTAS unless$\mathcal{P} = \mathcal{NP}$.*

It is noteworthy that the proof presented in this section does not require the graph to be directed, and directly yields the corresponding results also for undirected graphs. Note, however, that this does not hold for the proofs presented in Section 5.1.

6 | Restricting the Graph Class

Theorem 7 and 10 show that even highly restricted versions of the bi-objective minimum s-t-cut problem do not admit an MFPTAS. Hence, it is natural to ask whether exact algorithms or an MFPTAS can be obtained by restricting the graph class. In this section, we show that there exists a polynomial-time algorithm for MMCP under the following three assumptions:

Assumption 12.

(a) The tree-width k of the underlying undirected graph is constant,

(b) the number p of objectives is constant,[2] *and*

(c) the arc capacities u are integral and polynomially bounded, that is, each capacity $u(r)_i$, for $i \in \{1, \ldots, p\}$ and $r \in R$, is polynomial in the input size.

We note that all three assumptions are necessary. The bound on the tree-width is required since Theorem 6 shows that the problem is $\mathcal{NP}$-hard with just two objectives and binary capacities. If either the number of objectives or the arc capacities are not restricted, then the problem becomes intractable as shown in the following example.

Example 13. Let $G = (V, R)$ be the network illustrated in Figure 3, which features a series-parallel graph. Note that series parallel graphs have tree-width at most two, see [21]. In it, l is a positive integer and $V = \{s, v_1, \ldots, v_l, t\}$. Furthermore, for each $i \in \{1, \ldots, l\}$, R contains an arc from s to v_i and an arc from v_i to t.

To obtain an intractable example with polynomially bounded capacities, we use $2n$ objectives and assign the ith unit vector e_i to the arc from s to v_i and the $(l+i)$th unit vector e_{l+i} to the arc from v_i to t for each $i \in \{1, \ldots, l\}$.

For the case of a constant number of objectives, we use two objectives and assign capacity $(2^{i-1}, 0)$ to the arc from s to v_i and capacity $(0, 2^{i-1})$ to the arc from v_i to t for each $i \in \{1, \ldots, l\}$.

In both cases, is easy to see that all 2^l cuts in this network are efficient with pairwise different images, leading to a total of 2^l non-dominated points.

Hence, even if $\mathcal{P} = \mathcal{NP}$, there does not exist a polynomial-time exact algorithm for MMCP for graphs of tree-width at most two (and, in particular, for series-parallel graphs) when either the number of objectives or the capacity function is not restricted.

To obtain a polynomial-time exact algorithm for the case that all three assumptions in Assumption 12 hold, we start by introducing the concepts of tree-decompositions and tree-width (see, e.g., [22]) and then use the standard dynamic programming approach, which is illustrated in [23].

Definition 14. Let $G = (V, E)$ be a graph, $D = (V^D, E^D)$ a tree, and $\mathcal{V} = (V_d)_{d \in V^D}$ be a family of subsets $V_d \subseteq V$. Then,

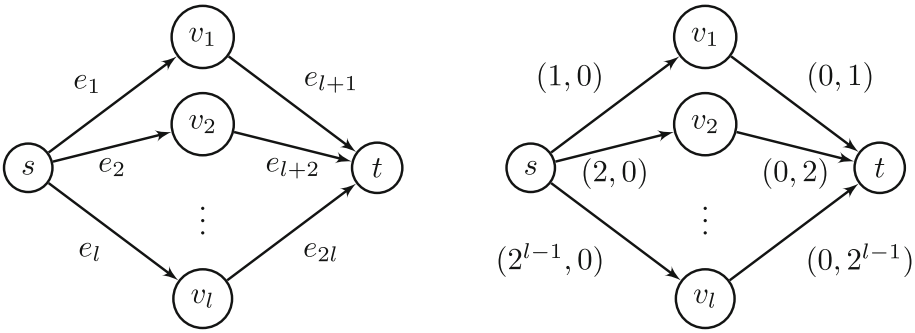

FIGURE 3 | Illustration of the networks from Example 13. The graph consists of vertices s, t, and $v_1, \ldots, v_l$, with arcs from s to each v_i and from each v_i to t. The labels on the arcs represent the capacities, where e_i denotes the ith unit vector.

$(D, \mathcal{V})$ is a *tree-decomposition of* G if it satisfies the following three conditions:

1. $V = \bigcup_{d \in V^D} V_d$,
2. for every edge $e \in E$, there exits a node $d \in V^D$ such that V_d contains both endpoints of e, and
3. for each vertex $v \in V$, the set $\{d \in V^D : v \in V_d\}$ induces a subtree of D.

The *width* of a tree-decomposition $(D, \mathcal{V})$ is defined as $\max\{|V_d| - 1 : d \in V^D\}$ and the *tree-width of* G is the minimum width of a tree-decomposition of G.

Note that we adopt the usual convention of using *vertex* when referring to the graph G and *node* when referring to the tree D.

Given a graph $G = (V, E)$ of tree-width k, it is known that a tree-decomposition $(D, \mathcal{V})$ of width k can be constructed in $\mathcal{O}(2^{k^3}|V|)$ time [24]. We may also assume that the number of nodes in a tree-decomposition is linear, that is, $|V^D| \in \mathcal{O}(|V|)$.

The following special type of tree-decomposition is algorithmically useful:

Definition 15. A tree-decomposition $(D, \mathcal{V})$ of a graph G is called *nice* if a root $r \in V^D$ can be chosen such that, in the tree rooted at $r \in V^D$, each node $d \in V^D$ falls into one of the following four categories:

Leaf node	d has no children and $\lvert V_d\rvert = 1$.
Forget node	d has one child d' and $V_d = V_{d'} \setminus \{v\}$ for a vertex $v \in V_{d'}$.
Introduce node	d has one child d' and $V_d = V_{d'} \cup \{v\}$ for a vertex $v \notin V_{d'}$.
Join node	d has two children d_1, d_2 and $V_d = V_{d_1} = V_{d_2}$.

Note that the children of a node d in V^D are its neighbors that are not on the unique path from d to r in D. More generally, we say that a node $d' \in V^D$ is a *descendant* of $d \in V^D$ if the unique path from r to d' passes through d. In particular, a node is its own descendant.

Theorem 16 ([25, Theorem 2.2.2]). *Let G be a graph and $(D, \mathcal{V})$ be a tree-decomposition of width k with $D = (V^D, E^D)$. Then G has a nice tree-decomposition of the same width with $\mathcal{O}(k \cdot |V^D|)$ nodes. Such a decomposition can be computed in $\mathcal{O}(k^2|V^D|)$ time.*

Therefore, given a graph $G = (V, E)$ of tree-width k, a nice tree decomposition with $\mathcal{O}(k|V|)$ nodes can be constructed in time $\mathcal{O}(2^{k^3}|V|)$, which is linear if k is constant. For more information on tree-decompositions, we refer to [22, 25, 26].

We now present our polynomial-time exact algorithm for MMCP under Assumption 12. Let $(D, \mathcal{V})$ be a nice tree-decomposition of G of width k, which exists by Theorem 16. We define

$$\begin{aligned} \mathrm{desc}_d &:= \{d' \in V^D : d' \text{ is a descendant of } d \text{ in } D\}, \\ X_d &:= \bigcup_{d' \in \mathrm{desc}_d} V_{d'}, \text{ and} \\ G_d &:= G[X_d]. \end{aligned}$$

Here, $G[X_d]$ denotes the subgraph of G induced by the nodes in X_d.

We now process D starting at the leaves and compute, at each node $d \in V^D$, a set $\mathrm{cut}_d(S)$ of cuts of G_d for each set $S \subseteq V_d$. For convenience, we write T for $V_d \setminus S$ when discussing the node d. The cuts (S', T') in $\mathrm{cut}_d(S)$ satisfy the following three invariants:

1. $S' \cap V_d = S$,
2. $\{s\} \cap X_d \subseteq S' \subseteq X_d \setminus \{t\}$, and
3. for each cut $C'' = (S'', T'')$ of G_d that satisfies Invariants (1) and (2), there exists a cut $C' \in \mathrm{cut}_d(S)$ with $\mathrm{cap}^{G_d}(C') \leqq \mathrm{cap}^{G_d}(C'')$.

We also store the capacity values with the cuts to avoid recomputation.

With access to these sets, we can obtain the desired non-dominated cuts by checking the set $\mathrm{cut}(G) := \bigcup_{S \subseteq V_r} \mathrm{cut}_r(S)$. Since $X_r = V$, Invariant (2) ensures that all cuts $(S', T') \in \mathrm{cut}(G)$ are s-t-cuts, while Invariant (3) guarantees that all efficient cuts are contained in $\mathrm{cut}(G)$. Thus, we just need to remove the dominated ones.

To compute the sets $\mathrm{cut}_d(S)$, we look at the four types of nodes in the nice tree-decomposition. Leaf nodes d are particularly easy, since for these G_d consists of a single vertex v and all cuts have value 0. Hence, we can usually simply set $\mathrm{cut}_d(\emptyset) := \{(\emptyset, \{v\})\}$ and $\mathrm{cut}_d(\{v\}) := \{(\{v\}, \emptyset)\}$, which satisfies Invariants (1) and (3). The only exceptions are the cases where $v = s$ or $v = t$, where we must instead set $\mathrm{cut}_d(\emptyset) := \emptyset$ and $\mathrm{cut}_d(\{t\}) := \emptyset$, respectively, in order to satisfy Invariant (2).

Next, consider a forget node d with child d' and $V_d = V_{d'} \setminus \{v\}$. Here, $X_d = X_{d'}$ and for $S \subseteq V_d$, we set $\mathrm{cut}_d(S) := \mathrm{cut}_{d'}(S) \cup \mathrm{cut}_{d'}(S \cup \{v\})$, where the cut values remain the same. Hence, all cuts added to $\mathrm{cut}_d(S)$ satisfy Invariants (1) and (2). To see that Invariant (3) holds, let $C'' = (S'', T'')$ be a cut in G_d with $S'' \cap V_d = S$ and $\{s\} \cap X_d \subseteq S'' \subseteq X_d \setminus \{t\}$. This is also a cut in $G_{d'}$ that satisfies Invariant (2) and $S'' \cap V_{d'}$ is either S or $S \cup \{v\}$. Thus, by Invariant (3) for d', $\mathrm{cut}_{d'}(S)$ or $\mathrm{cut}_{d'}(S \cup \{v\})$ contains a cut C' with $\mathrm{cap}^{G_{d'}}(C') \leqq \mathrm{cap}^{G_{d'}}(C'')$. Since these cuts are added to $\mathrm{cut}_d(S)$, the resulting collection of cuts satisfies Invariant (3). We also note that we may remove dominated cuts from the sets $\mathrm{cut}_d(S)$, which reduces their size without influencing the desired properties.

Introduce nodes are the third node type. Let d be such a node, with child d' and $V_d = V_{d'} \cup \{v\}$. Here, the cut values must be adjusted since $G_{d'} = G_d - v$ (where $G_d - v$ denotes $G[X_d \setminus \{v\}]$). For a fixed set $S \subseteq V_d$, we compute

$$u^+ := \begin{cases} u(R(v, T)) & \text{if } v \in S, \\ u(R(S, v)) & \text{if } v \notin S \end{cases}$$

and usually set

$$\mathrm{cut}_d(S) := \{(S' \cup S, T' \cup T) : (S', T') \in \mathrm{cut}_{d'}(S \setminus \{v\})\},$$

where we increment the capacity of each cut by u^+. Note that the notation $(S' \cup S, T' \cup T)$ just means that we add v to the side that should contain it. Just like in the leaf node case, if s is introduced, then we set $\mathrm{cut}_d(S) := \emptyset$ if $s \notin S$, and if t is introduced, we set $\mathrm{cut}_d(S) := \emptyset$ if $t \in S$.

To see that this is correct, we first note that the cuts we added to $\mathrm{cut}_d(S)$ satisfy Invariant (1) and (2). To verify Invariant (3), let $C'' = (S'', T'')$ be a cut in G_d with $S'' \cap V_d = S$ and $\{s\} \cap X_d \subseteq S'' \subseteq X_d \setminus \{t\}$. In particular, if either s or t is equal to the introduced node v, then we know that $s \in S$ or $t \notin S$ since such cuts do not exist otherwise.

Let $\tilde{C}'' := (S'' \setminus \{v\}, T'' \setminus \{v\})$ be the cut restricted to $G_{d'}$. Assume that $v \in S''$. Then $\mathrm{cap}^{G_d}(C'') = \mathrm{cap}^{G_{d'}}(\tilde{C}'') + u(R(v, T)) = \mathrm{cap}^{G_{d'}}(\tilde{C}'') + u^+$. Similarly, if $v \in T''$, then $\mathrm{cap}^{G_d}(C'') = \mathrm{cap}^{G_{d'}}(\tilde{C}'') + u(R(S, v)) = \mathrm{cap}^{G_{d'}}(\tilde{C}'') + u^+$. An analogous computation shows that extending a cut in $G_{d'}$ to one in G_d changes its capacity by exactly u^+. Hence, the capacities are updated correctly.

Moreover, by Invariant (3) for d', $\mathrm{cut}_{d'}(S \setminus \{v\})$ contains a cut $\tilde{C}'$ with $\mathrm{cap}^{G_{d'}}(\tilde{C}') \leq \mathrm{cap}^{G_{d'}}(\tilde{C}'')$. By extending the cut C' to G_d (by putting v on the appropriate side), we get

$$\mathrm{cap}^{G_d}(C') = \mathrm{cap}^{G_{d'}}(\tilde{C}') + u^+ \leq \mathrm{cap}^{G_{d'}}(\tilde{C}'') + u^+ = \mathrm{cap}^{G_t}(C'')$$

and Invariant (3) holds at d, too.

We now only need to deal with a join node d with children d_1 and d_2. In this case, $X_d = X_{d_1} \cup X_{d_2}$, and we set, for $S \subseteq V_d$,

$$\mathrm{cut}_d(S) = \{(S_1 \cup S_2, T_1 \cup T_2) : (S_1, T_1) \in \mathrm{cut}_{d_1}(S) \text{ and } (S_2, T_2) \in \mathrm{cut}_{d_2}(S)\}$$

If u_1 is the capacity of (S_1, T_1) in G_{d_1} and u_2 the capacity of (S_2, T_2) in G_{d_2}, then we obtain the capacity of $(S_1 \cup S_2, T_1 \cup T_2)$ in the set above as $u_1 + u_2 - u(R(S, T))$.

Again, we need to verify that this is correct. Invariants (1) and (2) hold by definition, and we note that $G_{d_1} \cap G_{d_2} = G[V_d]$ and there are no arcs between $G_{d_1} - V_d$ and $G_{d_2} - V_d$ by the properties of the tree-decomposition. Thus, an arc in the cut is also in the cut in G_{d_1} or G_{d_2}, and if both of its endpoints are in V_d, then it contributes to both, which is why we need to subtract $u(R(S, T))$ to avoid double counting. Thus, the cuts added to $\mathrm{cut}_d(S)$ have correctly computed capacities.

To complete the case of join nodes, we need to verify that Invariant (3) holds at d. Let $C'' = (S'', T'')$ be a cut in G_d and let $C_1'' = (S'' \cap X_{d_1}, T'' \cap X_{d_1})$ and $C_2'' = (S'' \cap X_{d_2}, T'' \cap X_{d_2})$ be the restrictions to G_{d_1} and G_{d_2} respectively. Then, by Invariant (3) for d_1 and d_2, there exist cuts $C_1' \in \mathrm{cut}_{d_1}(S)$ and $C_2' \in \mathrm{cut}_{d_2}(S)$ such that $\mathrm{cap}^{G_{d_1}}(C_1') \leqq \mathrm{cap}^{G_{d_1}}(C_1'')$ and $\mathrm{cap}^{G_{d_2}}(C_2') \leqq \mathrm{cap}^{G_{d_2}}(C_2'')$. If we extend these cuts back to G_d (by unifying the sides), we obtain a cut C' that is added to $\mathrm{cut}_d(S)$ and satisfies

$$\begin{aligned}\mathrm{cap}^{G_d}(C') &= \mathrm{cap}^{G_{d_1}}(C_1') + \mathrm{cap}^{G_{d_2}}(C_2') - u(R(S, T)) \\ &\leqq \mathrm{cap}^{G_{d_1}}(C_1'') + \mathrm{cap}^{G_{d_2}}(C_2'') - u(R(S, T)) \\ &= \mathrm{cap}^{G_d}(C''),\end{aligned}$$

which is what we wanted to prove.

Thus, we can solve MMCP using the proposed method and it only remains to consider the required running time. Let G be a graph with n vertices and tree-width k. By [24] and Theorem 16, we can compute a nice tree-decomposition of G of tree-width k in $\mathcal{O}(2^{k^3} n)$ time and this decomposition has $\mathcal{O}(kn)$ many nodes.

At a node d in the decomposition, the algorithm now needs to compute the sets $\mathrm{cut}_d(S)$. There are at most 2^k potential sets S and we need to bound the number of elements in each of these sets. To do so, we set $U_i := u(R)_i$ for $i \in \{1, \ldots, p\}$ and $U := (U_1 + 1) \cdot \ldots \cdot (U_p + 1)$. Since the value of the cut in the ith objective is somewhere in $\{0, \ldots, U_i\}$, cuts in $\mathrm{cut}_d(S)$ can obtain at most U distinct values. In particular, since we may eliminate dominated cuts, we never need to store more than U cuts in the set $\mathrm{cut}_d(S)$. By storing these cuts in an array (with an entry for each objective value), we can discard cuts of same value in constant time.

Therefore, the processing time ...

- of a leaf node is constant,
- of a forget node is in $\mathcal{O}(2^k U)$, to fill a new array for each set,
- of an introduce node is in $\mathcal{O}(2^k(U + n))$, where the additional n is caused by the computation of the u^+,
- of a join node is in $\mathcal{O}(2^k(U^2 + n))$. Here, the n comes from the computation of $u(R(S, T))$, and the U^2 stems from the fact that all pairs of cuts from the two children need to be considered.

Hence, we obtain the following theorem:

Theorem 17. *Let $G = (V, R, u)$ be a network of tree-width k and let $U_i := u(R)_i$ for $i \in \{1, \ldots, p\}$ and $U := (U_1 + 1) \cdots (U_p + 1)$. Then, MMCP can be solved in $\mathcal{O}(2^{k^3} n + kn2^k(U^2 + n))$ time. In particular, MMCP is solvable in quadratic time under Assumption 12.*

Since series-parallel graphs have tree-width at most two [21], Theorem 17 implies the following result:

Corollary 18. *If the network $G = (V, R, u)$ is series-parallel, the number of objectives is constant, and the arc capacities u are integral and polynomially bounded, then MMCP is solvable in polynomial time.*

Regarding the budgeted minimum s-t-cut problem, we obtain the following result:

Corollary 19. *If the network $G = (V, R, u)$ has constant tree-width and the arc capacities u and removal costs $b^{(i)}(r)$ are integral and polynomially bounded, the budgeted minimum s-t-cut*

problem can be solved in polynomial time if the number of budget constraints is constant.

7 | Conclusion

Motivated by the unique complexity status of the multi-objective minimum s-t-cut problem, this article explores its polynomial-time solvability and the existence of an MFPTAS under restrictions on the capacities and the graph class. To show the equivalence of polynomial-time solvability and the existence of an MFPTAS, we provide a general result for multi-objective problems with non-negative integer objective values that are polynomially bounded in the input size. Next, hardness results are shown for the case of two objectives and capacities in $\{(1,0),(0,1)\}$ or $\{(0,1),(1,1)\}$, which, from a more holistic point of view, implies that there is no restriction of the capacities in a way that none of the two objectives becomes obsolete such that the problem becomes polynomial-time solvable. This, in particular, yields the first multi-objective problem for which no MFPTAS exists even though the number of non-dominated points is linear in the input size and the single-objective version is polynomially solvable. Furthermore, our results yield improved hardness results for the single-objective budgeted minimum s-t-cut problem. We complement our hardness results by showing that, on graphs of bounded tree-width, the multi-objective minimum s-t-cut problem with polynomially bounded arc capacities can be solved exactly in polynomial time for any constant number of objectives.

While our results further emphasize the unique complexity status of the multi-objective minimum s-t-cut problem, a natural question is whether further relevant problems with similar properties exist. Moreover, since the best known approximation algorithms for the p-objective minimum s-t-cut problem yield $(p+\varepsilon)$-approximations, there is still a gap concerning the best obtainable approximation quality for the problem.

Author Contributions

Jan Boeckmann: investigation; writing – original draft; writing – review and editing; conceptualization; methodology; validation; formal analysis; project administration. **Stephan Helfrich:** methodology; conceptualization; investigation; validation; formal analysis; writing – review and editing; writing – original draft. **Oliver Bachtler:** conceptualization; methodology; investigation; validation; formal analysis; writing – review and editing; writing – original draft; visualization. **Stefan Ruzika:** writing – review and editing; supervision; validation; investigation; formal analysis. **Clemens Thielen:** investigation; validation; formal analysis; supervision; writing – review and editing.

Acknowledgments

Open Access funding provided by Johannes Kepler Universitat Linz/KEMÖ.

Funding

The Co-Author Oliver Bachtler was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - GRK 2982, 516090167 "Mathematics of Interdisciplinary Multiobjective Optimization".

Disclosure

The authors have nothing to report.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Endnotes

[1] The assumption of non-negative objectives is required for the definition of approximations and used throughout the corresponding literature, see [9].

[2] Note that the assumption that the number of objectives is constant is common in multi-objective optimization.

References

1. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows* (Prentice Hall, 1993).

2. S. R. Chestnut and R. Zenklusen, "Hardness and Approximation for Network Flow Interdiction," *Networks* 69, no. 4 (2017): 378–387, https://doi.org/10.1002/net.21739.

3. O. Goldschmidt and D. S. Hochbaum, "A Polynomial Algorithm for the k-Cut Problem for Fixed k," *Mathematics of Operations Research* 19, no. 1 (1994): 24–37, https://doi.org/10.1287/moor.19.1.24.

4. J. Boeckmann and C. Thielen, "A (B+1)-Approximation for Network Flow Interdiction With Unit Costs," *Discrete Applied Mathematics* 354 (2024): 58–71, https://doi.org/10.1016/j.dam.2021.07.008.

5. R. E. Gomory and T. C. Hu, "Multi-Terminal Network Flows," *Journal of the Society for Industrial and Applied Mathematics* 9, no. 4 (1961): 551–570, https://doi.org/10.1137/0109047.

6. M. Ehrgott, *Multicriteria Optimization*, vol. 491 (Springer Science & Business Media, 2005), https://doi.org/10.1007/3-540-27659-9.

7. M. Ehrgott, "Hard to Say It's Easy – Four Reasons Why Combinatorial Multiobjective Programmes Are Hard," in *Research and Practice in Multiple Criteria Decision Making: Proceedings of the XIVth International Conference on Multiple Criteria Decision Making (MCDM)* (Springer, 2000), 69–80, https://doi.org/10.1007/978-3-642-57311-8_5.

8. P. Serafini, "Some Considerations About Computational Complexity for Multi Objective Combinatorial Problems," in *Recent Advances and Historical Development of Vector Optimization: Proceedings of an International Conference on Vector Optimization* (Springer, 1987), 222–232, https://doi.org/10.1007/978-3-642-46618-2_15.

9. A. Herzel, S. Ruzika, and C. Thielen, "Approximation Methods for Multiobjective Optimization Problems: A Survey," *INFORMS Journal on Computing* 33, no. 4 (2021): 1284–1299, https://doi.org/10.1287/ijoc.2020.1028.

10. C. H. Papadimitriou and M. Yannakakis, "On the Approximability of Trade-Offs and Optimal Access of Web Sources," in *Proceedings 41st Annual Symposium on Foundations of Computer Science* (IEEE, 2000), 86–92, https://doi.org/10.1109/SFCS.2000.892068.

11. C. Glaßer, C. Reitwießner, H. Schmitz, and M. Witek, "Approximability and Hardness in Multi-Objective Optimization," in *Programs, Proofs, Processes: 6th Conference on Computability in Europe (CiE)* (Springer, 2010), 180–189, https://doi.org/10.1007/978-3-642-13962-8_20.

12. C. Bazgan, S. Ruzika, C. Thielen, and D. Vanderpooten, "The Power of the Weighted Sum Scalarization for Approximating Multiobjective Optimization Problems," *Theory of Computing Systems* 66, no. 1 (2022): 395–415, https://doi.org/10.1007/s00224-021-10066-5.

13. P. Halffmann, S. Ruzika, C. Thielen, and D. Willems, "A General Approximation Method for Bicriteria Minimization Problems," *Theoretical Computer Science* 695 (2017): 1–15, https://doi.org/10.1016/j.tcs.2017.07.003.

14. C. Bentz, M.-C. Costa, N. Derhy, and F. Roupin, "Cardinality Constrained and Multicriteria (Multi) Cut Problems," *Journal of Discrete Algorithms* 7, no. 1 (2009): 102–111, https://doi.org/10.1016/j.jda.2008.04.004.

15. H. Aissi and A. R. Mahjoub, "On the Minimum s-t Cut Problem With Budget Constraints," *Mathematical Programming* 203, no. 1 (2024): 421–442, https://doi.org/10.1007/s10107-023-01987-9.

16. S. Kratsch, S. Li, D. Marx, M. Pilipczuk, and M. Wahlström, "Multi-Budgeted Directed Cuts," *Algorithmica* 82 (2020): 2135–2155, https://doi.org/10.1007/s00453-019-00609-1.

17. D. Wojtczak, "On Strong NP-Completeness of Rational Problems," in *Proceedings of the 13th International Computer Science Symposium in Russia (CSR)*. LNCS, vol. 10846 (Springer, 2018), 308–320, https://doi.org/10.1007/978-3-319-90530-3_26.

18. G. J. Woeginger, "When Does a Dynamic Programming Formulation Guarantee the Existence of a Fully Polynomial Time Approximation Scheme (FPTAS)?," *INFORMS Journal on Computing* 12, no. 1 (2000): 57–74, https://doi.org/10.1287/ijoc.12.1.57.11901.

19. J. Chen and I. A. Kanj, "Constrained Minimum Vertex Cover in Bipartite Graphs: Complexity and Parameterized Algorithms," *Journal of Computer and System Sciences* 67, no. 4 (2003): 833–847, https://doi.org/10.1016/j.jcss.2003.09.003.

20. M. R. Garey, D. S. Johnson, and L. Stockmeyer, "Some Simplified NP-Complete Problems," in *Proceedings of the Sixth Annual ACM Symposium on Theory of Computing (STOC)* (Association for Computing Machinery, 1974), 47–63, https://doi.org/10.1145/800119.803884.

21. A. Brandstädt, V. B. Le, and J. P. Spinrad, *Graph Classes: A Survey*. SIAM Monographs on Discrete Mathematics and Applications, vol. 3 (SIAM, 1999), https://doi.org/10.1137/1.9780898719796.

22. H. L. Bodlaender, "A Partial k-Arboretum of Graphs With Bounded Treewidth," *Theoretical Computer Science* 209, no. 1 (1998): 1–45, https://doi.org/10.1016/S0304-3975(97)00228-4.

23. H. L. Bodlaender and A. M. C. A. Koster, "Combinatorial Optimization on Graphs of Bounded Treewidth," *Computer Journal* 51, no. 3 (2007): 255–269, https://doi.org/10.1093/comjnl/bxm037.

24. H. L. Bodlaender, "A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth," *SIAM Journal on Computing* 25, no. 6 (1996): 1305–1317, https://doi.org/10.1137/S0097539793251219.

25. C. Satzinger, *On Tree-Decompositions and Their Algorithmic Implications for Bounded-Treewidth Graphs*, Ph.D. thesis (Technische Universität Wien, 2014).

26. R. Diestel, *Graph Theory*, Graduate Texts in Mathematics, vol. 173, 5th ed. (Springer, 2016), https://doi.org/10.1007/978-3-662-53622-3.