Forcing a unique minimum spanning tree and a unique shortest path[*]

Tatsuya Gima[1] , Yasuaki Kobayashi[1] , Yota Otachi[2] , and Takumi Sato[1]

[1] Hokkaido University `gima@ist.hokudai.ac.jp`, `koba@ist.hokudai.ac.jp`,
`sato.takumi.h1@elms.hokudai.ac.jp`
[2] Nagoya University `otachi@nagoya-u.jp`

Abstract. A *forcing set* S in a combinatorial problem is a set of elements such that there is a unique solution that contains all the elements in S. An *anti-forcing set* is the symmetric concept: a set S of elements is called an anti-forcing set if there is a unique solution disjoint from S. There are extensive studies on the computational complexity of finding a minimum forcing set in various combinatorial problems, and the known results indicate that many problems would be harder than their classical counterparts: the decision version of finding a minimum forcing set for perfect matchings is NP-complete [Adams et al., Discret. Math. 2004] and that of finding a minimum forcing set for satisfying assignments for 3CNF formulas is Σ_2^{P}-complete [Hatami-Maserrat, DAM 2005]. In this paper, we investigate the complexity of the problems of finding minimum forcing and anti-forcing sets for the shortest s-t path problem and the minimum weight spanning tree problem. We show that, unlike the aforementioned results, these problems are tractable, with the exception of the decision version of finding a minimum anti-forcing set for shortest s-t paths, which is NP-complete.

Keywords: Forcing sets, minimum spanning trees, shortest paths

1 Introduction

This paper focuses on the problem of uniquely determining a solution by prescribing a subset of the solution (or of its complement). This concept has been extensively studied for various combinatorial problems, under several different names, such as (anti-)forcing number [AHM04; AMM04; HKŽ91; HSV07; Zha+25], critical sets [GHM05], and defining sets [HM05].

Harary, Klein, and Živković [HKŽ91] formulated the *forcing number* (for perfect matchings) of a graph, which is the smallest cardinality of an edge subset such that there is exactly one perfect matching including it. Prior to this work, the concept has been studied in chemistry as the innate degree of freedom of a Kekulé structure [KR87] and gained attention through several studies [AHM04; AMM04; Zha+25], including a recent survey article [Zha+25] on the forcing number and on its "dual" notion, the anti-forcing number [DZ17]. From the computational perspective, deciding whether the forcing number of an input graph is at most k is known to be NP-complete even on bipartite graphs [AHM04]. Similarly, given a bipartite graph G and its perfect matching M, it is NP-complete to decide whether there is a subset $S \subseteq M$ of size at most k such that every perfect matching M' of G other than M does not include S entirely (i.e., $S \setminus M' \neq \varnothing$) [AMM04]. Deng and Zhang [DZ17] proved that, given a bipartite graph G and its perfect matching, it is NP-complete to decide whether there is a subset $S \subseteq E(G) \setminus M$ of size at most k such that M is the unique perfect matching in $G - S$. In a more general context, this problem can be viewed as finding a smallest partial solution that is uniquely extended to a (complete) solution. Viewed from this broader perspective, the computational complexity of such problems has been investigated for various combinatorial problems, such as graph colorings [HM05], satisfiability [Dem+16; HM05], puzzles [Dem+16; KKF18], and the minimum vertex cover problem [An+25; Hor+24; Hor+25]. Table 1 shows several known results. An important observation is that the computational complexity of (most of) these problems would be harder than that of their classical setting.

Table 1: A summary of known and our results. [3]

	classical setting	forcing model	anti-forcing model
PERFECT MATCHING	P	NP-complete [AHM04]	unknown
3SAT	NP-complete	Σ_2^{P}-complete [HM05]	—
2SAT	P	NP-complete [Dem+16]	—
VERTEX COLORING	NP-complete	Σ_2^{P}-complete [HM05]	—
VERTEX COVER	NP-complete	Σ_2^{P}-complete [Hor+24]	
BIPARTITE VERTEX COVER	P	NP-complete [Hor+24]	
SHORTEST s-t PATH	P	P (Thm. 1)	NP-complete (Thm. 2)
MINIMUM SPANNING TREE	P	P (Thm. 5)	

We study the computational complexity of the problems of finding a minimum (anti-)forcing set for well-known tractable combinatorial problems, namely the shortest s-t path problem and the minimum-weight spanning tree problem. More specifically, let G be a directed graph with $s, t \in V(G)$ and let $w \colon E(G) \to \mathbb{R}_+$ be an edge-weight function. We say that $S \subseteq E(G)$ is a *forcing set for shortest s-t paths* if there is exactly one shortest path P from s to t in (G, w) such that $S \subseteq E(P)$. Similarly, $S \subseteq E(G)$ is an *anti-forcing set for shortest s-t paths* if there is exactly one shortest path P from s to t in (G, w) such that $S \cap E(P) = \varnothing$. Our first two problems are stated formally as follows.

> **Problem:** FORCINGSP / ANTI-FORCINGSP
> **Input:** A weighted digraph G with $w \colon E(G) \to \mathbb{R}_+$, $s, t \in V(G)$, and $k \in \mathbb{N}$.
> **Goal:** Decide whether there is a forcing set / anti-forcing set $S \subseteq E(G)$ with $|S| \leq k$ for shortest s-t paths in (G, w).

Similarly, we can define forcing and anti-forcing sets for minimum weight spanning trees. They are defined as follows.

> **Problem:** FORCINGMST / ANTI-FORCINGMST
> **Input:** A weighted graph G with $w \colon E(G) \to \mathbb{R}$.
> **Goal:** Decide whether there is a forcing set / anti-forcing set $S \subseteq E(G)$ with $|S| \leq k$ for minimum weight spanning trees in (G, w).

Let us note that the weight of an edge can be negative in the latter two problems, while the former two problems require the weight to be positive.

It is not hard to see that these four problems belong to NP, as we can check whether in polynomial time, given an edge set S, there is an optimal solution including or avoiding it. This motivates us to investigate whether these problems are NP-complete or polynomial-time solvable.

For FORCINGMST and ANTI-FORCINGMST, we can naturally generalize these problems to the settings in matroids. This generalization allows us to handle these two problems in a uniform way. In particular, we observe that the problem of finding a minimum forcing set for minimum weight bases in a matroid M is equivalent to that of finding a minimum weight anti-forcing set for maximum weight bases in the dual matroid M^*. We design a polynomial-time algorithm for computing a minimum anti-forcing set for maximum weight bases in a matroid M, assuming that M is given as an independence oracle (see Section 2 for de-

---

[3] Since the solutions of 3SAT, 2SAT, and VERTEX COLORING do not form subsets, these results cannot be directly regarded as results in the forcing model. However, since these problems deal with unique extensions of partial assignments on subsets of variables or vertices, it is still possible to interpret them as results in the forcing model. (See the discussion in [HM05].)

tails). This yields polynomial-time algorithms for FORCINGMST and ANTI-FORCINGMST (Theorem 5). Our polynomial-time algorithm exploits a well-known greedy algorithm for the maximum weight basis problem.

For FORCINGSP and ANTI-FORCINGSP, we can observe an intriguing distinction between them: FORCINGSP is polynomial-time solvable (Theorem 1), while ANTI-FORCINGSP is NP-complete (Theorem 2). To the best of our knowledge, this is the first example in which there is a complexity gap between the two models (see Table 1). Our polynomial-time algorithm for FORCINGSP is based on the simple reduction that reduces the problem on general weighted directed graphs to that on unweighted directed acyclic graphs. This reduction enables us to design a dynamic programming algorithm. For ANTI-FORCINGSP, we give a polynomial-time reduction from VERTEX COVER in Theorem 2. To overcome this intractability, we show that ANTI-FORCINGSP can be solved in linear time on graphs of bounded treewidth via Courcelle's theorem [CM93] (Theorem 4).

2 Preliminaries

Let E be a finite set. A *property* is a subset $\Pi \subseteq 2^E$. For $X \in \Pi$, a set $S \subseteq E$ is called a *forcing set for X* if it holds that $S \subseteq X$ and $S \not\subseteq X'$ for every $X' \in \Pi \setminus \{X\}$. In other words, S is a forcing set of X if X is the unique set in Π that contains S. Similarly, for $X \in \Pi$, a set $S \subseteq E$ is called an *anti-forcing set for X* if it holds that $S \cap X = \varnothing$ and $S \cap X' \neq \varnothing$ for every $X' \in \Pi \setminus \{X\}$. A set $S \subseteq E$ is called a *forcing set* (resp. *anti-forcing set*) *for property* $\Pi \subseteq 2^E$ if S is a forcing set (resp. anti-forcing set) for some $X \in \Pi$. We sometimes omit the target property Π when it is clear from the context.

For a (directed) graph G, we denote its vertex set by $V(G)$ and its edge set by $E(G)$. For an edge set $X \subseteq E(G)$, $G - X$ denotes the graph obtained from G by deleting all edges in X. Let G be a directed graph. For a vertex $v \in V(G)$, $N^-(v)$ denotes the set of in-neighbors $\{u : (u, v) \in E(G)\}$ of v. For a directed edge $e = (u, v) \in E(G)$, head(e) and tail(e) denote v and u, respectively.

Matroids. Let E be a finite set and let $\mathcal{I} \subseteq 2^E$. A pair $M = (E, \mathcal{I})$ is called a *matroid* if $\mathcal{I}$ is nonempty; for $X \in \mathcal{I}$, every subset of X belongs to $\mathcal{I}$; and for $X, Y \in \mathcal{I}$ with $|Y| > |X|$, $Y \setminus X$ has an element e such that $X \cup \{e\} \in \mathcal{I}$. Each set in $\mathcal{I}$ (resp. not in $\mathcal{I}$) is said to be *independent* (resp. *dependent*) in M. A maximal independent set is called a *basis* and a minimal dependent set is called a *circuit* in M. Every matroid has the following *symmetric basis-exchange property* [Bru69]: for any two bases A and B and for $a \in A \setminus B$, there is $b \in B \setminus A$ such that both $(A \cup \{b\}) \setminus \{a\}$ and $(B \cup \{a\}) \setminus \{b\}$ are bases. The collection of all bases of M is denoted by $\mathcal{B}(M)$. A *loop* in M is an element $e \in E$ such that $\{e\}$ is dependent.

Proposition 1 (Corollary 1.2.6 in [Oxl11]). *Let $B \in \mathcal{B}(M)$ and $e \notin B$. Then, $B \cup \{e\}$ contains a unique circuit C of M. Moreover, we have $(B \cup \{e\}) \setminus \{e'\} \in \mathcal{B}(M)$ for $e' \in C$.*

Let $M = (E, \mathcal{I})$ be a matroid. It is known that the pair $M^* := (E, \mathcal{I}^*)$ with

$$\mathcal{I}^* = \{X \subseteq E : X \subseteq E \setminus B \text{ for some } B \in \mathcal{B}(M)\}$$

forms a matroid [Oxl11], which is called the *dual matroid* of M. Note that $(M^*)^* = M$ for any matroid M. For $X \subseteq E$, we define pairs

$$M \mid X = (X, \{I \subseteq X : I \in \mathcal{I}\}),$$
$$M / X = (E \setminus X, \{I \subseteq E \setminus X : \exists B \in \mathcal{B}(M \mid X) \text{ s.t. } B \cup I \in \mathcal{I}\}).$$

These pairs are called the *restriction* and the *contraction* of M with respect to X, respectively. It is known that these pairs are also matroids [Oxl11] for any $X \subseteq E$. In this paper, a matroid M is given as an *independence oracle*, that is, we can query the oracle to decide whether a subset $X \subseteq E$ is independent in M.

Proposition 2 (Proposition 3.1.10 in [Oxl11]). *Let $X \subseteq E$. For every circuit C of M with $C \not\subseteq X$, there is a circuit C^* of M / X such that $C^* \subseteq C \setminus X$.*

Let $G = (V, E)$ be a multigraph. A subset of edges is said to be *acyclic* if it induces a forest in G. Let $\mathcal{F} \subseteq 2^E$ be the collection of all acyclic edge subsets of G. Then, it is known that the pair $M = (E, \mathcal{F})$ forms a matroid, which is called a *graphic matroid* [Oxl11].

3 Forcing a unique shortest path

In this section, we discuss FORCINGSP and ANTI-FORCINGSP. Recall that in these problems we are given a directed graph G with $s, t \in V(G)$ and an edge-weight function $w \colon E \to \mathbb{R}_+$. The numbers of vertices and edges in G are denoted by n and m, respectively. A forcing (resp. anti-forcing) set $S \subseteq E(G)$ (for shortest s-t paths) is *minimal* if any proper subset of S is not a forcing (resp. anti-forcing) set. It is easy to observe that for any minimal (anti-)forcing set S, every edge in S is contained in a shortest s-t path in G. This would reduce our problems to the cases where G is acyclic. More specifically, we let $d_s \colon V(G) \to \mathbb{R}_+ \cup \{0, \infty\}$ be the distance labeling from s in G, that is, $d_s(v)$ is the (shortest) distance from s to v in (G, w) for $v \in V(G)$. We remove all the edges $e = (u, v)$ that do not satisfy $d_s(v) = d_s(u) + w(e)$, which are edges that do not belong to any shortest path from s to v. We also remove all vertices (and their incident edges) that are not reachable from s or not reachable to t. As $w(e) > 0$ for each $e \in E$, the graph G' obtained in this way is indeed acyclic, which can be computed by a standard shortest path algorithm in $\mathcal{O}(m + n \log n)$ time. The following observation immediately follows from the fact that each shortest s-t path in (G, w) is an s-t path in G', and vice versa.

Observation 1. *Let $S \subseteq E$. Then, S is a minimal forcing set in G if and only if it is a minimal forcing set in G'. Similarly, S is a minimal anti-forcing set in G if and only if it is a minimal anti-forcing set in G'.*

This observation offers several advantages: we can assume that the input directed graph G is acyclic and all the paths from s to t are shortest paths in G, making the subsequent discussions simple.

3.1 Forcing set

In this subsection, we describe an algorithm for computing a minimum forcing set for shortest s-t paths. By Observation 1, we can assume that the given graph G is acyclic. Moreover, we can ignore the weight of edges, as every path from s to t in G is a shortest s-t path in the original graph.

For $v \in V(G)$, let $\mathcal{P}_v$ be the set of all paths from s to v in G. For vertices $u, v \in V(G)$, we write $u \to_{=1} v$ if there is exactly one path from u to v in G. This relation $\to_{=1}$ is reflexive, i.e., $v \to_{=1} v$ holds for any $v \in V(G)$. We assume that s has out-degree at least 2 since otherwise we can contract the (unique) out-going edge from s without affecting the solution. We define OPT$[e]$ and OPT$[v]$ as

$$\text{OPT}[e] := \min \{|S| : S \text{ is a forcing set for } \mathcal{P}_v \text{ with } e \in S\}$$

for $e = (u, v) \in E(G)$ and

$$\text{OPT}[v] := \min_{u \in N^-(v)} \text{OPT}[(u, v)]$$

for $v \in V(G) \setminus \{s\}$, while OPT$[s] := 0$. From now on, we show how to compute the values of OPT and then a minimum forcing set for $\mathcal{P}_t$ from these values.

The following lemma gives a characterization of a forcing set for a specific path $P \in \mathcal{P}_v$.

Lemma 1. *Let $S \subseteq E(G)$ and let $P \in \mathcal{P}_v$ be a path from s to v in G such that all edges in S are contained in P. Let $e_1, \dots, e_k$ be the edges in S appearing in this order on P. For $1 \leq i \leq k$, we let $e_i = (t_i, s_i)$, and let $s_0 = s$ and $t_{k+1} = v$. Then, S is a forcing set for P if and only if $s_i \to_{=1} t_{i+1}$ for all $0 \leq i \leq k$.*

Proof. Suppose that S is a forcing set for P. If there are at least two paths from s_i to t_{i+1} for some i, we can conclude that there is at least one path $P' \in \mathcal{P}_v$ with $P' \neq P$ that contains all the edges in S, which contradicts the uniqueness of P.

4

Conversely, suppose that $s_i \to_{=1} t_{i+1}$ for all $0 \le i \le k$. If S is not a forcing set for P, there is another path $P' \in \mathcal{P}_v$ such that all the edges in S are contained in P'. As $P \ne P'$, at least one edge of P is not contained in P'. We can assume that this edge appears on the subpath P_i of P between s_i and t_{i+1}. Since P' also passes though both s_i and t_{i+1}, the subpath of P' between them is distinct from P_i, contradicting $s_i \to_{=1} t_{i+1}$. $\qquad \square$

Corollary 1. *The minimum size of a forcing set for $\mathcal{P}_t$ is equal to*

$$\min_{v \in V(G)} \{ \mathrm{OPT}[v] : v \to_{=1} t \}.$$

We turn to a polynomial-time algorithm to compute the values of OPT. This immediately yields a polynomial-time algorithm for FORCINGSP due to Corollary 1.

Lemma 2. *For every $e = (u, v) \in E(G)$, it holds that*

$$\mathrm{OPT}[e] = \min_{w \in V(G)} \{ \mathrm{OPT}[w] : w \to_{=1} u \} + 1.$$

Proof. Let $S \subseteq E(G)$ be a forcing set for $P \in \mathcal{P}_v$ such that $|S| = \mathrm{OPT}[e]$ and $e \in S$. By Lemma 1, there is a vertex w' in P, which is either s or the head of an edge in $S \setminus \{e\}$ with $w' \to_{=1} u$. Observe that $S \setminus \{e\}$ is a forcing set for the subpath of P between s and w', as otherwise there are two paths from s to v including all edges in S, contradicting the uniqueness of P. Thus, we have

$$|S| = |S \setminus \{e\}| + 1 \ge \mathrm{OPT}[w'] + 1 \ge \min_{w \in V(G)} \{ \mathrm{OPT}[w] : w \to_{=1} u \} + 1.$$

Conversely, let w be a vertex minimizing $\mathrm{OPT}[w]$ under the condition that $w \to_{=1} u$ holds. Suppose first that $\mathrm{OPT}[w] = 0$. By definition, we have $w = s$. Since $s \to_{=1} u$, there is a unique path from s to u, which is denoted by P_{su}. By concatenating P_{su} and e in this order, we have a path P from s to v in G. Observe that there is exactly one path from s to v passing through e as $s \to_{=1} u$ and every path containing e must pass through u. Thus, $\{e\}$ is a forcing set for P, implying that $\mathrm{OPT}[e] \le \mathrm{OPT}[w] + 1$.

Suppose otherwise. In this case, $\mathrm{OPT}[w] = \mathrm{OPT}[e']$ for some edge e' incoming to w. Let S' be a forcing set for $P_{sw} \in \mathcal{P}_w$ such that $|S'| = \mathrm{OPT}[w]$ and $e' \in S'$. Similarly to the above case, the path obtained by concatenating P_{sw}, P_{wu}, and e in this order is the unique path P from s to v containing all edges in $S' \cup \{e\}$. Thus, $S' \cup \{e\}$ is a forcing set for P, implying that $\mathrm{OPT}[e] \le \mathrm{OPT}[w] + 1$. $\qquad \square$

Now, we describe our dynamic programming algorithm to compute the values of OPT. We first decide whether $u \to_{=1} v$ holds for each pair of vertices $u, v \in V(G)$. This can be done in total time $\mathcal{O}(nm)$ for all vertex pairs in G. We can evaluate $\mathrm{OPT}[e]$ and $\mathrm{OPT}[v]$ for each $e \in E(G)$ and each $v \in V(G)$ in time $\mathcal{O}(n)$, assuming that these values are evaluated in a dynamic programming manner. By Corollary 1, we can compute a minimum forcing set for $\mathcal{P}_t$ in time $\mathcal{O}(nm)$.

It is easy to extend our algorithm to compute a minimum forcing set for a specific path P by only evaluating $\mathrm{OPT}[e]$ and $\mathrm{OPT}[v]$ for each edge e and vertex v on the path P. Therefore, we have the following theorem.

Theorem 1. *FORCINGSP can be solved in time $\mathcal{O}(nm)$. Moreover, when additionally given a shortest s-t path P in G, we can compute a minimum forcing set for P in time $\mathcal{O}(nm)$ as well.*

It is not hard to see that, applying a standard trace-back technique, we can find a minimum forcing set for shortest s-t paths within the same running time. We would like to mention that our algorithm also works for undirected graphs since Observation 1 also holds for undirected graph G.

3.2 Anti-forcing set

We next consider the complexity of ANTI-FORCINGSP. In contrast to FORCINGSP, ANTI-FORCINGSP is NP-complete even for undirected and unweighted graphs.

Theorem 2. *ANTI-FORCINGSP is NP-complete even for undirected and unweighted graphs.*

Proof. We perform a polynomial-time reduction from VERTEX COVER. Recall that VERTEX COVER asks whether, given an undirected graph G and an integer k, there exists a vertex set $S \subseteq V$ of size at most k such that any edge $e \in E(G)$ is incident with at least one vertex in S. VERTEX COVER is known to be NP-complete [GJ79]. Our reduction produces a weighted multigraph, which can easily turn into an unweighted simple graph.

Construction. Let $\langle G, k \rangle$ be an instance of VERTEX COVER. Let $n = |V(G)|$. Now we give a construction of the reduction. See Fig. 1a for an example of the construction. Let $V(G) = \{v_1, \ldots, v_n\}$.

For each vertex v_i, we construct a vertex gadget H_i as follows. The gadget H_i contains a path of four vertices t_i, x_i, y_i, s_i, where the vertices have neighbors a_i, w_i, z_i, b_i, respectively. We add four paths of length 2 between a_i and b_i. The four internal vertices are denoted by $c_i^1, \ldots, c_i^4$.

We construct the entire multigraph H as follows. Let $N = 3n + k$. First, we add two special vertices $s = s_0$ and $t = t_{n+1}$. Then, we connect s_i and t_{i+1} by an edge for all $0 \leq i \leq n$. For each $1 \leq i \leq n$, we add $N + 2$ parallel edges between s and w_i and between z_i and t. For each edge $\{v_i, v_j\} \in E(G)$ with $i < j$, we add $N + 2$ parallel edges between y_i and x_j. We may regard and refer to these $N + 2$ parallel edges between two vertices as a single "thick edge". The weights of edges are defined as: $w(\{x_i, y_i\}) = 2$; $w(\{y_i, x_j\}) = 5(j - i - 1) + 3$; $w(\{s, w_i\}) = 5(i - 1) + 1$; and $w(\{z_i, t\}) = 5(n - i) + 1$ for all $i < j$; all the other edges are of weight 1. The graph constructed in this way is denoted by H. In this construction, every edge belongs to a shortest path (of length $5n + 1$) between s and t. Moreover, every shortest s-t path in H goes from the left to the right in Fig. 1.

We show that G has a vertex cover of cardinality at most k if and only if H has an anti-forcing set of cardinality at most N for shortest s-t paths. To simplify notation, we may refer to an edge $\{p_i, q_i\} \in E(H_i)$ as $(pq)_i$ and to a path with vertices $p_i, q_i, \ldots, r_i \in V(H_i)$ as $(pq \ldots r)_i$.

The only-if direction: Assume that G has a vertex cover $S \subseteq V(G)$ such that $|S| \leq k$. We define the edge set F as follows. If $v_i \in S$ then F includes $(xy)_i, (ac^2)_i, (ac^3)_i, (ac^4)_i$, otherwise F includes $(ta)_i, (wx)_i, (yz)_i$. Since $|S| \leq k$, the cardinality of F is at most $4|S| + 3|V(G) \setminus S| \leq 4k + 3(n - k) = N$. Let P be an s-t path in H that includes the subpath $(tac^1bs)_i$ if $v_i \in S$; includes the subpath $(txys)_i$ otherwise. Since P is a shortest s-t path (of length $5n + 1$) that does not contain any edge in F, it suffices to show that F is an anti-forcing set for P. Observe that for each i, either $(xy)_i \in F$ or $\{(wx)_i, (yz)_i\} \subseteq F$. Moreover, for each edge $\{v_i, v_j\} \in E(G)$, at least one of $(xy)_i$ and $(xy)_j$ is contained in F as S is a vertex cover of G. Thus, every s-t path that contains a thick edge is hit by F. This implies that P contains both t_i and s_i for each $1 \leq i \leq n$. For each i, there is exactly one path between t_i and s_i in $H_i - F$, which is the subpath of P. Hence, P is the unique path between s and t in $H - F$.

The if direction: Let F be an anti-forcing set with cardinality at most N and let P be the unique shortest s-t path in $H - F$. Since $|F| \leq N$ and there are $N + 2$ parallel edges as a thick edge, $H - F$ contains at least two of them: Due to the uniqueness of P, none of those parallel edges are contained in P. This implies that P contains both t_i and s_i for each $1 \leq i \leq n$. Hence, P contains either $(txys)_i$ or $(tac^jbs)_i$ as a subpath for each $1 \leq i \leq n$. Suppose that P contains $(txys)_i$. As $H - F$ contains both thick edges $\{s, w_i\}$ and $\{z_i, t\}$, F contains both $(wx)_i$ and $(yz)_i$. Moreover, due to the uniqueness of P, F contains at least one edge of the path forming $(tac^jbs)_i$ for all j. Thus, F contains at least three edges in H_i. Suppose otherwise that P contains $(tac^jbs)_i$ for some $1 \leq j \leq 4$. Similarly to the above case, F contains at least one edge on the path $(txys)_i$. Moreover, as there are four edge-disjoint paths between a_i and b_i, F contains at least three edges between them. Thus, F contains at least four edges in H_i in this case. Let $S \subseteq V$ be the set of all vertices v_i such that P contains $(tac^jbs)_i$ as a subpath for some j. Since F contains at most $N = 3n + k$ edges,
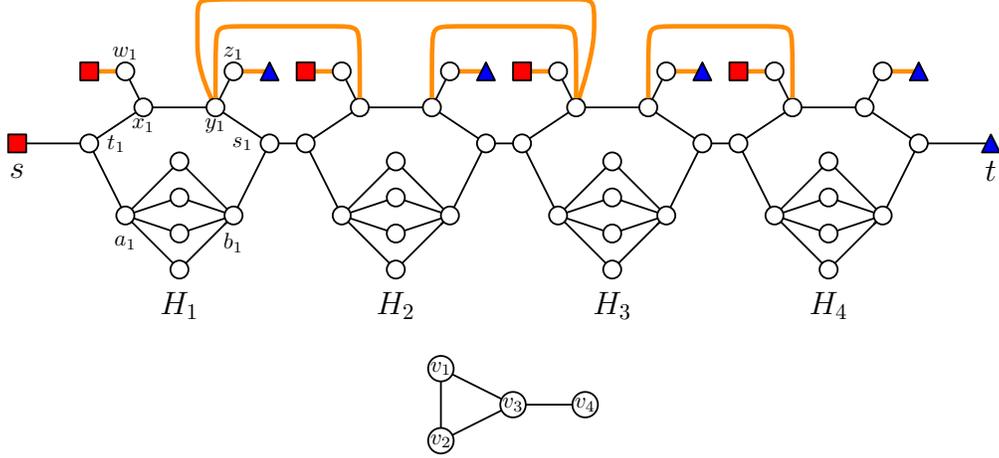
(a) The lower graph is the original instance $\langle G, k \rangle$, and the upper graph is the constructed instance H.
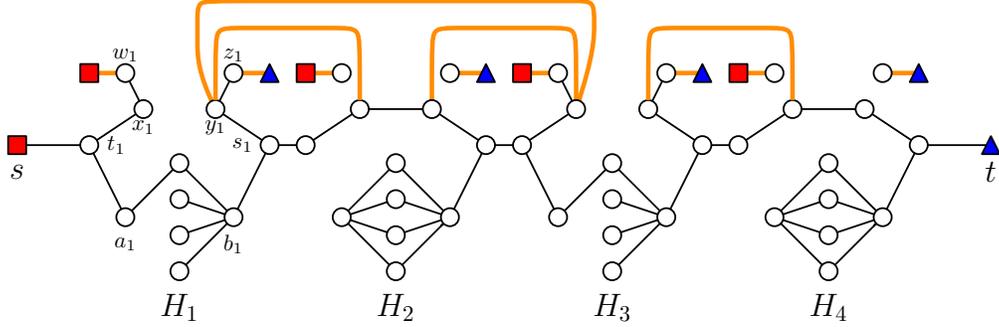
(b) Illustration of $H - F$, where F is the anti-forcing set constructed from a vertex cover $\{v_1, v_3\}$ of G.

Fig. 1: Example of the reduction from VERTEX COVER to ANTI-FORCINGSP in the proof of Theorem 2. For an aesthetic reason, the figure contains multiple copies of vertices representing the same vertex: all red boxes correspond to the vertex s and all blue triangles correspond to the vertex t. Thick orange edges represent "thick edges", which are parallel edges between them.

S contains at most k vertices. For each edge $\{v_i, v_j\} \in E(G)$ with $i < j$, P cannot contain both subpaths $(txys)_i$ and $(txys)_j$, which make a bypass passing through the thick edge $\{y_i, x_j\}$. This yields that S covers all edges in G.

Making H simple and unweighted: Finally, let us note that H can be converted into a simple graph with uniform edge weight. We can replace an edge of weight $q \in \mathbb{N}$ with a path of length q. Moreover, we can remove parallel edges by replacing all edges in H (including non-parallel edges) with paths of length 2. It is easy to see that these replacements preserve the size of a smallest anti-forcing set of H. Since the weight of each edge is $\mathcal{O}(n)$ and the multiplicity of each thick edge is $\mathcal{O}(n)$, the size of the resulting graph is upper bounded by a polynomial in n, completing our reduction. □

While ANTI-FORCINGSP is NP-complete, we can find in polynomial time a minimum cost anti-forcing set for a given shortest s-t path P in G by reducing it to the minimum multiway cut problem on directed acyclic graphs, which can be solved in polynomial time [Ben07].

Theorem 3. *Let G be an edge-weighted directed graph with $s, t \in V(G)$ and let P be a shortest path from s to t in G. Then, a minimum anti-forcing set S for P can be computed in polynomial time.*

Proof. We reduce the problem to MULTIWAY CUT on directed acyclic graphs. Let H be a directed graph and let $T \subseteq V(H)$ be a terminal set. A *multiway cut* for T is a set $S \subseteq E(H)$ such that for any terminal pair $s, t \in T$, there is no path from s to t in $H - S$. Given a directed graph H and $T \subseteq V(H)$, the task of MULTIWAY CUT is to compute a minimum cardinality multiway cut of (H, T).

Due to Observation 1, we can assume that the given graph G is a directed acyclic graph and the paths from s to t are exactly the shortest s-t paths in the original graph. We show that S is an anti-forcing set for s-t paths of G if and only if S is a multiway cut for $V(P)$ on $H := G - E(P)$.

Suppose that $S \subseteq E(G)$ is an anti-forcing set for P, that is, P is the unique path from s to t with $S \cap E(P) = \varnothing$. Suppose that there is a path P' from $u \in V(P)$ to $v \in V(P)$ in H. We then obtain an s-t path $P'' (\neq P)$ from P by replacing the subpath between u and v with P'. Since S is an anti-forcing set, S contains at least one edge of P'', and thus, $E(P') \cap S \neq \varnothing$. Hence, S is a multiway cut for $V(P)$ in H.

Conversely, suppose that S is a multiway cut for $V(P)$ in H. Since every path from u to v for any distinct $u, v \in V(P)$ in H contains an edge in S, P is the unique path from s to t in $G - S$, meaning that S is an anti-forcing set for P.

It is known that MULTIWAY CUT can be solved in polynomial time on directed acyclic graphs [Ben07]. Thus, a minimum anti-forcing set S for P can be computed in polynomial time as well. □

Finally, we observe that the problem can be solved in linear time on graphs of bounded treewidth. Since it is not necessary to discuss what treewidth is in the proof, we omit its definition in this paper (see e.g. [Cyg+15, Chapter 7]). We observe that ANTI-FORCINGSP is reduced to one for evaluating a monadic second-order (MSO_2) formula, and thus, by Courcelle's theorem [ALS91; CM93] and Bodlaender's algorithm [Bod96], it can be solved in linear time on graphs of bounded treewidth.

Theorem 4. *Let G be a directed graph with two specified vertices s and t such that the underlying graph of G has bounded treewidth. Then, a minimum anti-forcing set S for shortest s-t paths on G can be computed in linear time.*

Proof. Due to Observation 1, it is sufficient to compute, for directed acyclic graph G', a minimum edge set S such that $G' - S$ has a unique s-t path. We express this condition by an MSO_2-formula over directed graphs with two constant vertices s and t. In MSO_2 logic, we can quantify edges, vertices, vertex sets, and edge sets, and we can use the standard (logical) connectives '=', '∧', '¬', and so on (for a more formal definition, see e.g., [Cyg+15]). For edge variable $e = (u, v)$, we use $\mathtt{head}(e)$ and $\mathtt{tail}(e)$ that are functions returning v and u, respectively.

We use $\exists! X \; \phi(X)$ as an abbreviation for $\exists X \; \forall Y \; [X = Y \leftrightarrow \phi(Y)]$. Note that $\exists! X \; \phi(X)$ is uniqueness quantification, that is, it can be interpreted as meaning "there is exactly one X such that $\phi(X)$". Now, we consider the following MSO_2 formula[4]

$$\phi(X) \equiv X \subseteq E \wedge \exists! P \; (P \subseteq E \wedge P \cap X = \varnothing \wedge \mathtt{Path}_{s,t}(P)).$$

Here, $\mathtt{Path}_{s,t}(P)$ is a formula that expresses that a directed edge set P is an s-t path. It is folklore that the formula $\mathtt{Path}_{s,t}(P)$ can be defined by an MSO_2-formula, which is given below for completeness.

$$\mathtt{Path}_{s,t}(P) \equiv \mathtt{D2}_{s,t}(P) \wedge \forall P' \subseteq P \; (P = P' \vee \neg \mathtt{D2}_{s,t}(P')),$$

$$\mathtt{D2}_{s,t}(P) \equiv \begin{pmatrix} \exists! e \; (e \in P \wedge s = \mathtt{tail}(e)) \wedge \exists! f \; (f \in P \wedge t = \mathtt{head}(f)) \wedge \\ \forall e \in P \; \neg (s = \mathtt{head}(e) \vee t = \mathtt{tail}(e)) \wedge \\ \forall e \in P \; \begin{pmatrix} (\mathtt{head}(e) \neq t) \leftrightarrow \exists! f \; (f \in P \wedge \mathtt{head}(e) = \mathtt{tail}(f)) \wedge \\ (\mathtt{tail}(e) \neq s) \leftrightarrow \exists! f \; (f \in P \wedge \mathtt{tail}(e) = \mathtt{head}(f)) \end{pmatrix} \end{pmatrix}.$$

It is known that a set S that satisfies the property $\phi(S)$ defined by a constant size MSO_2-formula ϕ and minimizes its cardinality $|S|$ can be found in linear time on directed graphs such that the underlying graph has bounded treewidth [ALS91; Bod96; CM93]. Since the underlying graph G'' of G' is a subgraph of G, G'' has bounded treewidth. Thus, the claim holds. □

---

[4] We use abbreviated notations such as $\subseteq, \cap, = \varnothing, \forall x \in X \phi(X)$, but these are interpreted in the standard way and can be defined in MSO_2 logic.

Algorithm 1: An algorithm for ANTI-FORCINGMST.

   Input: A multigraph G with weight function $w : E(G) \to \mathbb{R}$.
   Output: A minimum anti-forcing set for minimum weight spanning trees of (G, w).
1 begin
2 $S \leftarrow \varnothing; T \leftarrow \varnothing$;
3   **while** $E(G) \neq \varnothing$ **do**
4 $w_{\min} \leftarrow \min_{e \in E(G)} w(e)$;
5 $E_{\min} \leftarrow \{e \in E(G) : w(e) = w_{\min}\}$;
6   Let $G_{\min}$ be the subgraph $(V(G), E_{\min})$ of G;
7   Let $F_{\min}$ be a maximal forest of $G_{\min}$;
8 $T \leftarrow T \cup F_{\min}$;
9 $S \leftarrow S \cup (E_{\min} \setminus (F_{\min} \cup \{e \in E_{\min} : e \text{ is a self-loop in } G_{\min}\}))$;
10   Let G be the graph obtained by contracting all edges in $E_{\min}$;
11   **return** S;

4 Forcing a unique minimum weight matroid basis

In this section, we give polynomial-time algorithms for FORCINGMST and ANTI-FORCINGMST. The basic strategy of both algorithms follows Kruskal's algorithm [Kru56] for computing a minimum weight spanning tree. We first briefly explain our algorithms for FORCINGMST and ANTI-FORCINGMST in Section 4.1 and then generalize them to the setting of matroids in Section 4.2.

4.1 Special case: Minimum weight spanning trees

First, we sketch an intuition behind our algorithm for ANTI-FORCINGMST. A formal discussion, including the correctness of the algorithm, is deferred to Section 4.2. Let G be a connected edge-weighted multigraph. Contracting an edge of G may create parallel edges or self-loops, and we do not remove them in the contraction operation. Let us recall Kruskal's algorithm for computing a minimum weight spanning tree of G: Starting from $T = \varnothing$, we repeatedly choose a minimum weight edge e of G that does not form a cycle with the previously chosen edges T and add it to T until it becomes a spanning tree of G. Instead of adding edges one by one, we can modify this procedure to add, in bulk, a maximal forest F consisting only of minimum weight edges $E_{\min}$ that have not been chosen yet. We repeatedly apply this to the graph obtained from G by contracting all edges in $E_{\min}$ as long as $E(G)$ is nonempty. It is easy to observe that this procedure also computes a minimum weight spanning tree T of G. Since the choice of a maximal forest F consisting of edges in $E_{\min}$ is not unique, the solution T obtained by this procedure is not unique as well. Thus, to force the solution to be unique, it is necessary to include all edges in $E_{\min} \setminus F$, except for self-loops, as an anti-forcing set. The algorithm that formalizes this intuition is given in Algorithm 1. For FORCINGMST, we can design a similar algorithm: Instead of including all non-loop edges in $E_{\min} \setminus F$, include all edges in F that are not bridges in $G_{\min}$. The pseudocode of this algorithm is given in Algorithm 2. It can be easily confirmed that both algorithms for ANTI-FORCINGMST and FORCINGMST run in time $\mathcal{O}(m \log n)$ using a standard analysis of Kruskal's algorithm and a linear-time algorithm for enumerating bridges [Tar74]. The proof of their correctness is deferred to the next subsection.

Theorem 5. *FORCINGMST and ANTI-FORCINGMST can be solved in time $\mathcal{O}(m \log n)$, where $n = |V(G)|$ and $m = |E(G)|$.*

4.2 General case: Matroid bases

In this subsection, we consider generalized versions of FORCINGMST and ANTI-FORCINGMST, where the goals are to find minimum forcing and anti-forcing sets for minimum weight bases of a matroid M. Clearly, FORCINGMST and ANTI-FORCINGMST are special cases, where M is a graphic matroid.

9

Algorithm 2: An algorithm for FORCINGMST.

Input: A multigraph G with weight function $w\colon E(G) \to \mathbb{R}$.
Output: A minimum forcing set for minimum weight spanning trees of (G, w).

1 begin
2 $S \leftarrow \varnothing; T \leftarrow \varnothing;$
3   **while** $E(G) \neq \varnothing$ **do**
4 $w_{\min} \leftarrow \min_{e \in E(G)} w(e);$
5 $E_{\min} \leftarrow \{e \in E(G) : w(e) = w_{\min}\};$
6   Let $G_{\min}$ be the subgraph $(V, E_{\min})$ of G;
7   Let $F_{\min}$ be a maximal forest of $G_{\min}$;
8 $T \leftarrow T \cup F_{\min};$
9 $S \leftarrow S \cup \{e \in F_{\min} : e \text{ is not a bridge of } G_{\min}\};$
10   Let G be the graph obtained by contracting all edges in $E_{\min}$;
11   **return** $S;$

Theorem 6. *Let $M = (E, \mathcal{I})$ be a matroid with weight function $w\colon E \to \mathbb{R}$. Assume that M is given as an independence oracle. Then, a minimum forcing set for minimum weight bases of (M, w) can be computed in polynomial time. Similarly, a minimum anti-forcing set for minimum weight bases of (M, w) can be computed in polynomial time as well.*

Let $\mathcal{B}_{\min}(M)$ and $\mathcal{B}_{\max}(M)$ be the collections of minimum and maximum weight bases of M, respectively. At first, we show that the problem of finding a minimum forcing set for $\mathcal{B}_{\min}(M)$ is equivalent to that of finding a minimum anti-forcing set for $\mathcal{B}_{\max}(M^*)$, where M^* is the dual matroid of M.

Observation 2. *A set $S \subseteq E$ is a forcing set for $\mathcal{B}_{\min}(M)$ if and only if S is an anti-forcing set for $\mathcal{B}_{\max}(M^*)$.*

Proof. Let $B \subseteq E$ be a minimum weight basis of (M, w) and $B^* = E \setminus B$. Note that B^* is a basis of the dual matroid M^*. Let $S \subseteq E$. Now, $S \subseteq B$ if and only if $S \setminus B = \varnothing$. Combined with $S \setminus (E \setminus B^*) = S \cap B^*$, we conclude that $S \subseteq B$ if and only if $S \cap B^* = \varnothing$. Since B is a minimum weight basis of (M, w) if and only if B^* is a maximum weight basis of (M^*, w), the claim holds. ☐

Observation 2 suggests the relations in Fig. 2, enabling us to find a minimum forcing set for $\mathcal{B}_{\min}(M)$ by applying an algorithm for finding a minimum anti-forcing set for $\mathcal{B}_{\max}(M^*)$. Note that a basis B minimizes the value $w(B)$ if and only if it maximizes $-w(B)$. Now, we are ready to describe our algorithm for computing
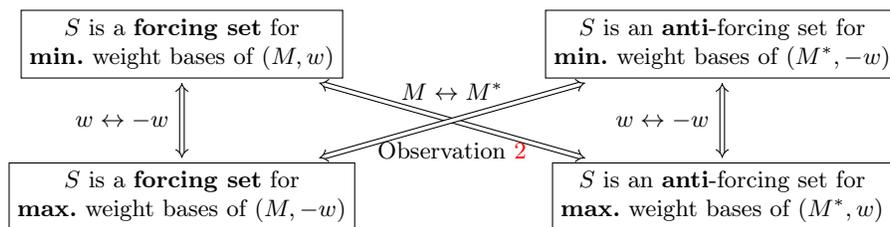

Fig. 2: The equivalence of the problems.

a minimum anti-forcing set for $\mathcal{B}_{\min}(M)$, which is shown in Algorithm 3. The underlying idea of the algorithm is analogous to Algorithm 1. At each iteration of the main loop, we consider the matroid $M_{\min}$ consisting only of the minimum weight elements $E_{\min}$, and select a basis $B_{\min}$ from $M_{\min}$. The crux of its correctness is that it is necessary and sufficient to include all the elements in $E_{\min} \setminus B_{\min}$ except for loops in any anti-forcing set for $\mathcal{B}_{\min}(M)$.

Algorithm 3: An algorithm for a minimum anti-forcing set for $\mathcal{B}_{\min}(M)$

Input: Matroid $M = (E, \mathcal{I})$ with weight function $w \colon E \to \mathbb{R}$.
Output: A minimum anti-forcing set for $\mathcal{B}_{\min}(M)$.
1 begin
2 $S \leftarrow \varnothing; B \leftarrow \varnothing$;
3   **while** $E \neq \varnothing$ **do**
4 $w_{\min} \leftarrow \min_{e \in E} w(e)$;
5 $E_{\min} \leftarrow \{e \in E : w(e) = w_{\min}\}$;
6 $M_{\min} \leftarrow M \mid E_{\min}$;
7   Let $B_{\min}$ be a basis of $M_{\min}$;
8 $B \leftarrow B \cup B_{\min}$;
9 $S \leftarrow S \cup (E_{\min} \setminus (B_{\min} \cup \{e \in E_{\min} : e \text{ is a loop in } M_{\min}\}))$;
10 $M \leftarrow M \,/\, E_{\min}$;
11   **return** S;

Lemma 3. *Algorithm 3 returns a minimum anti-forcing set for $\mathcal{B}_{\min}(M)$.*

Proof. First, we show that the output S of Algorithm 3 is an anti-forcing set for $\mathcal{B}_{\min}(M)$. From the correctness of the greedy algorithm for computing a minimum weight basis of a matroid, at the end of the main loop, we have $B \in \mathcal{B}_{\min}(M)$. Moreover, we have $B \cap S = \varnothing$.

Let $B' \in \mathcal{B}_{\min}(M)$ with $B' \neq B$. It suffices to show that $B' \cap S \neq \varnothing$. Now, there exists an element $e \in B' \setminus B$ since B and B' are distinct bases of M. We can assume that, at some iteration of the main loop, $e \in E_{\min}$ and $e \notin B_{\min}$ hold. Since e is not a loop in $M_{\min}$ and $e \in E_{\min} \setminus B_{\min}$, we have $e \in S$. Hence, $B' \cap S \neq \varnothing$. This concludes that S is an anti-forcing set for $\mathcal{B}_{\min}(M)$.

Next, we show that S is a minimum anti-forcing set for $\mathcal{B}_{\min}(M)$. Let S' be a minimum anti-forcing set for $\mathcal{B}_{\min}(M)$ and $B' \in \mathcal{B}_{\min}(M)$ be the unique minimum weight basis of M with $B' \cap S' = \varnothing$. Let w_i be the value of $w_{\min}$ in the i-th iteration of the while-loop and let $E_i = \{e \in E : w(e) = w_i\}$. For each i, let $E_{\leq i} = E_1 \cup \cdots \cup E_i$ and let $M_i = M \,/\, E_{\leq i-1}$. Note that $M_i \mid E_i = M_{\min}$ in the i-th iteration.

Claim. $|B \cap E_i| = |B' \cap E_i|$ holds for all i.

Proof. We prove a slightly stronger statement: for $B_1, B_2 \in \mathcal{B}_{\min}(M)$, it holds that $|B_1 \cap E_i| = |B_2 \cap E_i|$ for all i. Suppose otherwise. We choose $B_1, B_2 \in \mathcal{B}_{\min}(M)$ with $|B_1 \cap E_j| \neq |B_2 \cap E_j|$ for some j in such a way that the symmetric difference $|B_1 \triangle B_2| := |B_1 \setminus B_2| + |B_2 \setminus B_1|$ is minimized. We assume without loss of generality that $|B_1 \cap E_j| > |B_2 \cap E_j|$. Since $B_1 \neq B_2$, by the symmetric basis-exchange property, for $e_1 \in (B_1 \setminus B_2) \cap E_j$, there exists $e_2 \in B_2 \setminus B_1$ such that both $B'_1 := (B_1 \cup \{e_2\}) \setminus \{e_1\}$ and $B'_2 := (B_2 \cup \{e_1\}) \setminus \{e_2\}$ are bases in M. Since $B_1, B_2 \in \mathcal{B}_{\min}(M)$, we have $w(e_1) = w(e_2)$, and hence $e_1, e_2 \in E_j$. Moreover, $B'_1, B'_2 \in \mathcal{B}_{\min}(M)$. This contradicts the choice of B_1, B_2 as

$$|B'_1 \triangle B_2| = |((B_1 \cup \{e_2\}) \setminus \{e_1\}) \triangle B_2| = |B_1 \triangle B_2| - 2 < |B_1 \triangle B_2|$$

and $|B'_1 \cap E_j| = |B_1 \cap E_j| \neq |B_2 \cap E_j|$. ∎

Suppose to the contrary that $|S'| < |S|$. This implies that there is an index j such that $|S' \cap E_j| < |S \cap E_j|$. By the above claim, we have $|B \cap E_j| = |B' \cap E_j|$. Let $e \in E_j \setminus (B' \cup S')$ that is not a loop in $M_j \mid E_j$. We can choose such an element e since $|E_j| = |B \cap E_j| + |S \cap E_j| + |L_j|$, where L_j is the set of loops in $M_j \mid E_j$, and $|S' \cap E_j| < |S \cap E_j|$. Note that e is not a loop of M_j. Due to Proposition 1, $B' \cup \{e\}$ has a unique circuit C of M. If C contains an element $e' \in E \setminus E_{\leq j}$, then $(B' \cup \{e\}) \setminus \{e'\}$ is a basis of M with weight strictly smaller than B', contradicting $B' \in \mathcal{B}_{\min}(M)$. Thus, C consists of only elements in $E_{\leq j}$. If C contains an element $e' \in E_j$, the basis $(B' \cup \{e\}) \setminus \{e'\}$, which belongs to $\mathcal{B}_{\min}(M)$, avoids S', contradicting the uniqueness of B'. Hence, all the elements of C except for e belong to $E_{\leq j-1}$. As $C \nsubseteq E_{\leq j-1}$, by Proposition 2, there is a circuit C^* of M_j such that $C^* \subseteq C \setminus E_{\leq j-1}$. This circuit is indeed a singleton $C^* = \{e\}$, contradicting the fact that e is not a loop in M_j. □

11

Finally, we consider the running time of Algorithm 3. Given a matroid $M = (E, \mathcal{I})$ as an independence oracle and $X, Y \subseteq E$, we can decide whether Y is independent in $M \mid X$, in M / X, and in M^* with a polynomial number of oracle calls to M. Thus, each step of Algorithm 3 can be performed in polynomial time. Thus, Theorem 6 holds.

Similarly to Theorem 1, we can compute smallest forcing and anti-forcing sets for a given $B^* \in \mathcal{B}_{\min}(M)$ in polynomial time, by just taking $B_{\min}$ at line 7 in Algorithm 3 as $B_{\min} = B^* \cap E_{\min}$.

Corollary 2. *Let $M = (E, \mathcal{I})$ be a matroid with weight function $w : E \to \mathbb{R}$. Assume that M is given as an independence oracle. Given a minimum weight basis $B^* \in \mathcal{B}_{\min}(M)$, a minimum forcing set / a minimum anti-forcing set for B^* can be computed in polynomial time.*

References

[AHM04]   Peyman Afshani, Hamed Hatami, and Ebadollah S. Mahmoodian. "On the spectrum of the forced matching number of graphs". In: *Australas. J Comb.* 30 (2004), pp. 147–160.

[ALS91]   Stefan Arnborg, Jens Lagergren, and Detlef Seese. "Easy Problems for Tree-Decomposable Graphs". In: *J. Algorithms* 12.2 (1991), pp. 308–340. DOI: 10.1016/0196-6774(91)90006-K.

[AMM04]   Peter Adams, Mohammad Mahdian, and Ebadollah S. Mahmoodian. "On the forced matching numbers of bipartite graphs". In: *Discret. Math.* 281.1-3 (2004), pp. 1–12. DOI: 10.1016/J.DISC.2002.10.002.

[An+25]   Shinwoo An, Yeonsu Chang, Kyungjin Cho, O-joung Kwon, Myounghwan Lee, Eunjin Oh, and Hyeonjun Shin. "Pre-Assignment Problem for Unique Minimum Vertex Cover on Bounded Clique-Width Graphs". In: *39th AAAI Conference on Artificial Intelligence, AAAI 2025.* AAAI Press, 2025, pp. 26886–26894. DOI: 10.1609/AAAI.V39I25.34893.

[Ben07]   Cédric Bentz. "The maximum integer multiterminal flow problem in directed graphs". In: *Oper. Res. Lett.* 35.2 (2007), pp. 195–200. DOI: 10.1016/J.ORL.2006.03.005.

[Bod96]   Hans L. Bodlaender. "A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth". In: *SIAM J. Comput.* 25.6 (1996), pp. 1305–1317. DOI: 10.1137/S0097539793251219.

[Bru69]   Richard A. Brualdi. "Comments on bases in dependence structures". In: *Bulletin of the Australian Mathematical Society* 1.2 (1969), pp. 161–167. DOI: 10.1017/S000497270004140X.

[CM93]   Bruno Courcelle and Mohamed Mosbah. "Monadic Second-Order Evaluations on Tree-Decomposable Graphs". In: *Theoretical Computer Science* 109.1&2 (1993), pp. 49–82. DOI: 10.1016/0304-3975(93)90064-Z.

[Cyg+15]   Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. *Parameterized Algorithms.* Springer, 2015. DOI: 10.1007/978-3-319-21275-3.

[Dem+16]   Erik D. Demaine, Fermi Ma, Ariel Schvartzman, Erik Waingarten, and Scott Aaronson. "The Fewest Clues Problem". In: *8th International Conference on Fun with Algorithms, FUN 2016.* Vol. 49. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2016, 12:1–12:12. DOI: 10.4230/LIPICS.FUN.2016.12.

[DZ17]   Kai Deng and Heping Zhang. "Anti-forcing spectra of perfect matchings of graphs". In: *J. Comb. Optim.* 33.2 (2017), pp. 660–680. DOI: 10.1007/S10878-015-9986-3.

[GHM05]   Mahya Ghandehari, Hamed Hatami, and Ebadollah S. Mahmoodian. "On the size of the minimum critical set of a Latin square". In: *Discret. Math.* 293.1-3 (2005), pp. 121–127. DOI: 10.1016/J.DISC.2004.08.026.

[GJ79]   Michael R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[HKŽ91]   Frank Harary, D. J. Klein, and T. P. Živković. "Graphical properties of polyhexes: Perfect matching vector and forcing". In: *Journal of Mathematical Chemistry* 6 (1991), pp. 295–306. DOI: 10.1007/BF01192587.

[HM05]   Hamed Hatami and Hossein Maserrat. "On the computational complexity of defining sets". In: *Discret. Appl. Math.* 149.1-3 (2005), pp. 101–110. DOI: 10.1016/J.DAM.2005.03.004.

[Hor+24]   Takashi Horiyama, Yasuaki Kobayashi, Hirotaka Ono, Kazuhisa Seto, and Ryu Suzuki. "Theoretical Aspects of Generating Instances with Unique Solutions: Pre-assignment Models for Unique Vertex Cover". In: *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024*. AAAI Press, 2024, pp. 20726–20734. DOI: 10.1609/AAAI.V38I18.30060.

[Hor+25]   Takashi Horiyama, Kazuhisa Seto, Fumiya Sakamoto, and Ryu Suzuki. "Hardness of Pre-assignment Problem for Unique Minimum Vertex Cover on Planar Graphs with Maximum Degree 3". In: *25th International Symposium on Fundamentals of Computation Theory, FCT 2025*. Vol. 16106. Lecture Notes in Computer Science. 2025, pp. 238–251. DOI: 10.1007/978-3-032-04700-7_18.

[HSV07]   Frank Harary, Wolfgang Slany, and Oleg Verbitsky. "On the Computational Complexity of the Forcing Chromatic Number". In: *SIAM J. Comput.* 37.1 (2007), pp. 1–19. DOI: 10.1137/050641594.

[KKF18]   Kei Kimura, Takuya Kamehashi, and Toshihiro Fujito. "The Fewest Clues Problem of Picross 3D". In: *9th International Conference on Fun with Algorithms, FUN 2018*. Vol. 100. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2018, 25:1–25:13. DOI: 10.4230/LIPICS.FUN.2018.25.

[KR87]   D. J. Klein and M. Randić. "Innate degree of freedom of a graph". In: *Journal of Computational Chemistry* 8.4 (1987), pp. 516–521. DOI: 10.1002/jcc.540080432.

[Kru56]   Joseph B. Kruskal. "On the shortest spanning subtree of a graph and the traveling salesman problem". In: *Proc. Amer. Math. Soc.* (1956), pp. 48–50. DOI: 10.1090/S0002-9939-1956-0078686-7.

[Oxl11]   James G. Oxley. *Matroid theory*. Second. Oxford University Press, 2011.

[Tar74]   Robert Endre Tarjan. "A Note on Finding the Bridges of a Graph". In: *Inf. Process. Lett.* 2.6 (1974), pp. 160–161. DOI: 10.1016/0020-0190(74)90003-9.

[Zha+25]   Yaxian Zhang, Xin He, Qianqian Liu, and Heping Zhang. "Forcing, anti-forcing, global forcing and complete forcing on perfect matchings of graphs - A survey". In: *Discret. Appl. Math.* 376 (2025), pp. 318–347. DOI: 10.1016/J.DAM.2025.06.022.