# Strongly Polynomial and Fully Combinatorial Algorithms for Bisubmodular Function Minimization

S. Thomas McCormick [*] Satoru Fujishige [†]

August 6, 2007

**Abstract**

Bisubmodular functions are a natural "directed", or "signed", extension of submodular functions with several applications. Recently Fujishige and Iwata showed how to extend the Iwata, Fleischer, and Fujishige (IFF) algorithm for submodular function minimization (SFM) to bisubmodular function minimization (BSFM). However, they were able to extend only the weakly polynomial version of IFF to BSFM. Here we investigate the difficulty that prevented them from also extending the strongly polynomial version of IFF to BSFM, and we show a way around the difficulty. This new method gives a somewhat simpler strongly polynomial SFM algorithm, as well as the first combinatorial strongly polynomial algorithm for BSFM. This further leads to extending Iwata's fully combinatorial version of IFF to BSFM.

## 1 Introduction

We start by motivating our interest in bisubmodular function minimization via its connection to 0, $\pm 1$ integral linear systems. We then cover the history of its algorithms, and how the rest of the paper is organized.

### 1.1 Motivation

A key paradigm of combinatorial optimization is to model discrete optimization problems as integer programs. If we are lucky, it turns out that solving the relaxed LP version of our integer model produces an integral solution, and hence one that solves the integer program. Thus a big area of study is which classes of linear programs have guaranteed integral optimal solutions.

One useful class comes from LPs with totally unimodular matrices. However, this class is somewhat limited as it has been shown to contain essentially only network matrices and their duals (see, e.g., Schrijver [47, Section 19.4]).

A larger class is the set of totally dual integral (TDI) problems, (see, e.g., Schrijver [47, Chapter 22]). Despite often not being totally unimodular, these problems have integral optimal solutions. TDI systems often have size exponential in the natural variables of the model. Often

[*]Tom.McCormick@sauder.ubc.ca, Sauder School of Business, University of British Columbia, Vancouver, BC V6T 1Z2 Canada. Supported by an NSERC Operating Grant.

[†]fujishig@kurims.kyoto-u.ac.jp, Research Institute for Mathematical Sciences, Kyoto University, Kyoto 606-8502, Japan. Supported by a Grant-in-Aid of the Ministry of Education, Culture, Science and Technology of Japan.

it is possible to show a separation algorithm that then implies a polynomial algorithm via the Ellipsoid Method (see, e.g., Grötschel, Lovász, and Schrijver [27, 28]), but this is unsatisfactory as Ellipsoid-based algorithms are reputed to be slow in practice. We would prefer to have *combinatorial* (i.e., non-Ellipsoid) algorithms for these problems. Finding such combinatorial algorithms has been a big area of research in recent times.

One type of constraint that often arises is $Ax \le b$ when $A$ is 0–1. In such cases, let $E$ index the columns of $A$. For each row $i$, $S_i = \{j \in E \mid a_{ij} = 1\} \subseteq E$, and we could consider the RHS $b_i$ as a function of $S_i$, call it $r(S_i)$. If we further denote the family of subsets induced by the rows by $\mathcal{R}$, then our constraints look like

$$\sum_{e \in S} x_e \equiv x(S) \le r(S) \quad \text{for all } S \in \mathcal{R}. \tag{1}$$

Then it is interesting to ask under what conditions on $\mathcal{R}$ and $r$ this system is TDI.

A very important class of instances where (1) is TDI is when $\mathcal{R}$ is closed under unions and intersections (is a *ring family*), and when $r$ is submodular on $\mathcal{R}$, i.e.,

$$r(S) + r(T) \ge r(S \cup T) + r(S \cap T) \quad \text{for all } S,\, T \in \mathcal{R}.$$

As usual we identify $S \subseteq E$ with its incidence vector $\chi^S$, where $\chi^S_e = 1$ if $e \in S$, and $\chi^S_e = 0$ if $e \notin S$. In this notation $x(S) = (\chi^S)'x$. In this sense we can think of set union as the set of indices where $\chi^S + \chi^T$ is non-zero, and set intersection as the set of indices where $\chi^S + \chi^T$ has magnitude two. Note that $\chi^S + \chi^T = \chi^{S \cap T} + \chi^{S \cup T}$.

Suppose that (1) is part of a larger model, and we have a current point $\bar{x}$ that we want to check for feasibility to (1). Often $|\mathcal{R}|$ is exponential in $|E|$, and so brute force is not efficient. Instead we would like to solve $\min_{S \in \mathcal{R}}\{r(S) - \bar{x}(S)\}$. If the optimal value is non-negative, then $\bar{x}$ is feasible, and otherwise an optimal solution $S$ gives a violated constraint.

When $r$ is submodular, the set function $r(S) - \bar{x}(S)$ is again submodular. Hence this is a special case of the general problem

**Submodular Function Minimization (SFM):** $\min_{S \in \mathcal{R}} f(S)$,
where $f$ is submodular on ring family $\mathcal{R}$.

It turns out that SFM has many, many applications, see e.g., [32, 36]. Hence finding a combinatorial algorithm for SFM is important.

A natural next step from (1) is when matrix $A$ is 0, $\pm 1$. In this case, for row $i$ define $S_i^+ = \{j \in E \mid a_{ij} = +1\}$ and $S_i^- = \{j \in E \mid a_{ij} = -1\}$, and consider the RHS $b_i$ to be a function of the ordered pair $(S_i^+, S_i^-)$, call it $r(S^+, S^-)$. We call such an ordered pair of subsets $S = (S^+, S^-)$ where $S^+ \cap S^- = \emptyset$ a *signed set*. If we also denote the family of such signed sets by $\mathcal{R}$, then our system looks like

$$x(S) \equiv x(S^+) - x(S^-) \le r(S) \quad \text{for all } S \in \mathcal{R}. \tag{2}$$

Again it is interesting to ask under what conditions on $\mathcal{R}$ and $r$ this system is TDI. To answer this we need "signed" versions of union and intersection. This implicitly uses a bijection between signed sets $S = (S^+, S^-)$ and their signed incidence vectors $\chi^S_e = +1$ if $e \in S^+$, $-1$ if $e \in S^-$, and 0 otherwise, so that again $x(S) = (\chi^S)'x$. Since the incidence vectors have three possible values, we let $3^E$ denote the family of all signed sets on ground set $E$. We globally define $n = |E|$, so that $|3^E| = 3^n$.

If $S$, $T \in 3^E$, then the set of indices where $\chi^S + \chi^T$ is positive is $(S^+ \cup T^+) - (S^- \cup T^-)$, and the set of indices where $\chi^S + \chi^T$ is negative is $(S^- \cup T^-) - (S^+ \cup T^+)$, and so we define $S \sqcup T = ((S^+ \cup T^+) - (S^- \cup T^-), (S^- \cup T^-) - (S^+ \cup T^+))$. The set of indices where $\chi^S + \chi^T$ equals $+2$ is $S^+ \cap T^+$, and the set of indices where $\chi^S + \chi^T$ equals $-2$ is $S^- \cap T^-$, and so we define $S \sqcap T = (S^+ \cap T^+, S^- \cap T^-)$. This implies that $\chi^S + \chi^T = \chi^{S \sqcap T} + \chi^{S \sqcup T}$. We call $\mathcal{R} \subseteq 3^E$ a *signed ring family* if it is closed under $\sqcup$ and $\sqcap$ (see Fujishige [19] for a similar analysis of 0, $\pm 1$ vectors). Then a function $f$ from signed ring family $\mathcal{R}$ to $\mathbb{R}$ is called *bisubmodular* if it satisfies

$$f(S) + f(T) \geq f(S \sqcup T) + f(S \sqcap T) \quad \text{for all } S,\, T \in \mathcal{R}. \tag{3}$$

Kabadi and Chandrasekaran [34, 35] showed that (2) is TDI if $\mathcal{R}$ is a signed ring family and $f$ is bisubmodular on $\mathcal{R}$. The problem of deciding whether a current point $\bar{x}$ satisfies (2) again reduces to minimizing $f(S) - \bar{x}(S)$, again a bisubmodular function. Hence this is a special case of the general problem

**Bisubmodular Function Minimization (BSFM):** $\min_{S \in \mathcal{R}} f(S)$,
where $f$ is bisubmodular on signed ring family $\mathcal{R}$.

BSFM also has several other applications (see Section 2.3), and so finding a combinatorial algorithm for BSFM is important.

### 1.1.1 Why "Bisubmodular"?

Suppose that $f$ is bisubmodular on $3^E$. Then for each fixed $S$, $T \subseteq E$, $f(S, X)$ (for $X \subseteq E - S$) and $f(X, T)$ (for $X \subseteq E - T$) are submodular in $X$. Note that a bisubmodular function is not in general submodular. Hence (by analogy to "bilinear") "bisubmodular" is a good name for this property. However, it was called by other names in the past: "polypseudomatroid" [12], "universal polymatroid" [41], and "generalized submodularity" [49, Section 49.11.d].

Conversely, the name "bisubmodular" was used for a different concept described in Schrijver [49, Section 49.11.d]. That concept reduces to classic submodularity on 0, $\pm 1$ vectors where the $+1$s and $-1$s are restricted to a fixed partition of $E$. Fujishige [19] considers yet another notion that again reduces to submodularity on 0, $\pm 1$ vectors, but with no fixed partition.

## 1.2 History of SFM and BSFM Algorithms

The importance of SFM has been recognized since the early days of combinatorial optimization, and finding a polynomial algorithm for SFM was a long-standing open problem. In 1981 Grötschel, Lovász, and Schrijver [27] realized that Ellipsoid could be used to get a polynomial algorithm for SFM, and later [28] was able to extend this result to show how to use Ellipsoid to get a strongly polynomial algorithm for SFM. But finding a combinatorial SFM algorithm remained open.

Nearly simultaneously in 1999, two working papers appeared giving quite different combinatorial strongly polynomial algorithms for SFM. These were by Schrijver [48] (formally published in 2000) and Iwata, Fleischer, and Fujishige (IFF) [33] (formally published in 2001). We call the core weakly polynomial version of the IFF Algorithm just "IFF" and its strongly polynomial variant IFF-SP. Various improvements to these algorithms have since appeared, including a Push-Relabel speedup to Schrijver's Algorithm by Fleischer and Iwata [18] which we call FlI-PR;

an improved analysis of Schrijver's Algorithm by Vygen [51]; a "fully combinatorial" version of IFF by Iwata [30] that we call I-FC; and a speedup of the core subroutine of IFF incorporating ideas from Schrijver's Algorithm by Iwata [31] called the Hybrid Algorithm that speeds up IFF, IFF-SP, and I-FC. Recently Orlin [42] developed a combinatorial strongly polynomial SFM algorithm which somewhat resembles Schrijver's Algorithm, but with many new ideas, which is $O(n \log n)$ faster than the strongly polynomial version of Hybrid.

It has been relatively rare that a TDI problem such as SFM whose only known polynomial algorithm uses Ellipsoid makes the transition to having a combinatorial algorithm. Therefore it is tempting to speculate that the algorithmic techniques developed for the SFM combinatorial (weakly, strongly, fully combinatorial) polynomial algorithms can be extended to other problems. A natural place to start looking for such extensions is BSFM. The first polynomial algorithm for BSFM was given by Qi [43], but it was based on Ellipsoid. Fujishige and Iwata [24] found the first combinatorial BSFM algorithm by extending IFF to BSFM via an algorithm we call FuI-BSFM. However, they were unable to extend IFF-SP to BSFM (and I-FC came later). Hence it is natural to ask whether IFF-SP, I-FC, and the speedups of the Hybrid Algorithms can also be extended to BSFM.

The main results in this paper show how to modify IFF-SP so that it and I-FC extend to BSFM (so far we are not able to extend Hybrid). Furthermore, in order to derive our extension of IFF-SP we also show how to adapt these algorithms to directly deal with the case where the bisubmodular function is defined on a signed ring family, which extends techniques from [30].

### 1.3 Organization of the Paper

Section 2 covers various technical details about bisubmodularity, including applications, a graphical representation of signed ring families and how to deal with it algorithmically, generating vertices of the polyhedron using a Signed Greedy Algorithm, optimality conditions for BSFM, the method used by the algorithms to represent the current feasible point, and a useful estimate of the "size" of the current point.

Section 3 describes the new strongly polynomial (BSFM-SP) and fully combinatorial (BSFM-FC) algorithms. These algorithms are based in large part on the same ideas as the Fujishige and Iwata FuI-BSFM algorithm, so this section concentrates mostly on the parts that are different: how to handle the signed ring family, and the modifications needed to attain strongly polynomial and then fully combinatorial versions. After going over the basic framework of the algorithms in Sections 3.1 and 3.2, Section 3.4 develops BSFM-SP, and Section 3.5 develops BSFM-FC. Then Section 4 covers some extensions related to BSFM: finding *all* BSFM solutions, minimizing separable convex objectives on the bisubmodular polyhedron, and using BSFM to solve a line search problem over bisubmodular polyhedra. Finally, Section 5 summarizes and poses some open questions.

## 2 Background on Bisubmodularity

### 2.1 BSFM Preliminaries

We consider $E$ to be the base set of "unsigned" elements. When $e \in E$ occurs in some $T \in 3^E$ it must be either positive (in $T^+$) or negative (in $T^-$). We define $\overline{E}$ to be the set of *signed* elements, so that each $e \in E$ corresponds to two different members of $\overline{E}$, a positive element

$+e$ and a negative element $-e$. If $t \in \overline{E}$ is a signed element, then we use $\mathrm{abs}(t)$ to denote the corresponding unsigned element of $E$. We use $-t$ to denote the negation of $t$, so that for $e \in E$, $-(+e) = -e$ and $-(-e) = +e$. We extend these notions to subsets, so that if $S = (S^+, S^-) \in 3^E$, then $\mathrm{abs}(S)$ is the set of unsigned elements $\{\mathrm{abs}(s) \mid s \in S\}$ and $-S$ is the signed set $(S^-, S^+)$. For $t \in \overline{E}$ we use $\mathrm{sgn}(t)$ to denote the sign of $t$, so that if $t$ is positive, then $\mathrm{sgn}(t) = +$, and if $t$ is negative, then $\mathrm{sgn}(t) = -$. Therefore for any $t \in \overline{E}$, $t = \mathrm{sgn}(t)\mathrm{abs}(t)$. We consistently use letters $e$, $g$, $h$, …, $l$ for elements of $E$, and letters $p$, $q$, …, $u$ for elements of $\overline{E}$. Note that $S \subseteq \overline{E}$ need not be a signed set (since we could have $u$, $-u \in S$), but if $S$ contains at most one of $+e$, $-e$ for each $e \in E$, then we do consider $S$ to be a signed set in the natural way.

If $T \in 3^E$ and $q \in \overline{E}$ but $\mathrm{abs}(q) \notin \mathrm{abs}(T)$, then if $q \in \overline{E}$ is positive, $T + q$ is the set $(T^+ + \mathrm{abs}(q), T^-)$ (where $T^+ + \mathrm{abs}(q)$ stands for $T^+ \cup \{\mathrm{abs}(q)\}$); if $q$ is negative, then $T + q$ is the set $(T^+, T^- + \mathrm{abs}(q))$. If $t \in \overline{E}$, then $f(t)$ stands for $f(\{\mathrm{abs}(t)\}, \emptyset)$ if $t$ is positive, and $f(\emptyset, \{\mathrm{abs}(t)\})$ if $t$ is negative. We use $\emptyset$ to also stand for $(\emptyset, \emptyset)$. Note that if $q \in \overline{E}$ is positive, then $T = (T^+, T^-) \in 3^E$ contains $q$ iff $\mathrm{abs}(q) \in T^+$; if $\mathrm{abs}(q) \in T^-$ then we would write $q \notin T$.

For $S = (S^+, S^-)$, $T = (T^+, T^-) \in 3^E$ we write $S \sqsubseteq T$ if $S^+ \subseteq T^+$ and $S^- \subseteq T^-$. Then it is easy to show that (3) implies that

$$f(T+t) - f(T) \leq f(S+t) - f(S) \quad \text{for all } S \sqsubseteq T \in 3^E \text{ and } t \in \overline{E} \text{ such that } \mathrm{abs}(t) \notin \mathrm{abs}(T). \quad (4)$$

This is the bisubmodular equivalent to the familiar decreasing incremental cost characterization of submodularity. Bisubmodularity also directly implies that

$$f(T+t) + f(T+(-t)) \geq 2f(T) \quad \text{for all } t \in \overline{E} \text{ such that } \mathrm{abs}(t) \notin \mathrm{abs}(T). \qquad (5)$$

In fact [3] shows that (4) and (5) give an alternate characterization of bisubmodularity.

## 2.2 The Bisubmodular Polyhedron

If $f$ is bisubmodular on signed ring family $\mathcal{R}$, then the *bisubmodular polyhedron* is $P_{\mathcal{R}}(f) = \{y \in \mathbb{R}^E \mid y(S) \leq f(S) \text{ for all } S \in \mathcal{R}\}$. For our arguments to be consistent for every case we need to worry about the constraint $0 = y(\emptyset) \leq f(\emptyset)$. To ensure that this makes sense, from this point forward we re-define $f(S)$ to be $f(S) - f(\emptyset)$ so that $f(\emptyset) = 0$; note that this change does not affect bisubmodularity nor BSFM. It is known that $P_{\mathcal{R}}(f)$ is never empty.

A bisubmodular function $f$ can have as many as $3^n$ values, so even inputing all values of $f$ is exponential in $n$. Hence we use the standard assumption that $f$ is represented via an *evaluation oracle* $\mathcal{E}$, which is a black box whose input is some $S \in 3^E$, and whose output is $f(S)$. We use EO to denote the running time of $\mathcal{E}$, and separately count calls to $\mathcal{E}$ in our running times. We also use $M$ to denote a bound on the maximum absolute value of $f(S)$ over all $S \in 3^E$, a measure of the size of $f$.

## 2.3 BSFM Applications

Section 1.1 gives our first application: given a point $\bar{x} \in \mathbb{R}^E$, BSFM solve the separation problem of deciding whether or not $\bar{x} \in P_{\mathcal{R}}(f)$.

Given an instance of SFM with $f : \mathcal{R} \to \mathbb{R}$, define the signed ring family $\bar{\mathcal{R}} = \{(R, \emptyset) \mid R \in \mathcal{R}\}$. Then $f$ is bisubmodular on $\bar{\mathcal{R}}$, and BSFM for $f$ on $\bar{\mathcal{R}}$ is just SFM for $f$ on $\mathcal{R}$. Allowing a signed ring family makes embedding SFM into BSFM simpler than what is done in [24].

Delta-matroids were introduced by Bouchet [10] and Chandrasekaran and Kabadi [12], with a slightly restricted version considered by Dress and Havel [16]. The membership problem for the rank function of a delta-matroid is an instance of BSFM, and [24] was the first paper to give a combinatorial polynomial time algorithm for it. A particular case of delta-matroids is the convex hull of perfectly matchable node sets of an undirected graph, and here Cunningham and Green-Krótki [15] developed a combinatorial algorithm.

Delta-matroids are further extended to jump systems by Bouchet and Cunningham [11], such that convex hulls of jump systems are precisely integral bisubmodular polyhedra. One example is the $b$-matching degree sequence polyhedron of Cunningham and Green-Krótki [14]. Zhang [52] gives a combinatorial algorithm for membership in $b$-matching degree sequence polyhedra.

Note that many of these applications involve separation: given a point $\bar{x} \in \mathbb{R}^E$, decide whether $\bar{x} \in P_{\mathcal{R}}(f)$; if not, give a separating hyperplane proving it. In such cases $\bar{x}$ is likely to be highly fractional, and so having a strongly polynomial algorithm for BSFM is especially useful. In Section 4.3 we apply BSFM-FC to a line search problem over $P_{\mathcal{R}}(f)$, and that application requires having a fully combinatorial algorithm.

A natural further extension of BSFM is to minimizing a separable convex objective over a bisubmodular polyhedron, and there are applications that involve such non-linear objectives. Fujishige [20, Section 5] gives an algorithmic framework for solving this problem that depends on an oracle for computing general bisubmodular exchange capacities. In Section 4.2 we show how our BSFM algorithms can be used to compute these, and thereby give a complete algorithm for such problems.

Bisubmodularity also arises in bicooperative games, see Bilbao et al. [7]. In this context BSFM is useful for deciding if a given point belongs to the core of the game or not.

Finally, we point out a "non-application", where there is a constraint of the form $x(S^+) - x(S^-) \leq f(S^+, S^-)$ but which is not bisubmodular. Queyranne and Wang [44] consider the polyhedron associated with a precedence-constrained scheduling problem. The separation problem for their class of *serial constraints* is formally similar to the BSFM separation problem, but in general the sets of interest do not form a ring family, and the function is not bisubmodular.

## 2.4 Bisubmodularity on Signed Ring Families

FuI-BSFM works for $f$ defined over $3^E$. We develop our algorithms to work over signed ring families for two reasons: (1) As with SFM, some applications (such as the separation problem in Section 1) define $f$ only on such a sub-family, and (2) the strongly polynomial (and so also the fully combinatorial) version of our algorithm, which we call BSFM-SP (and BSFM-FC) generates subproblems over signed ring families.

We must start with some sort of representation of $\mathcal{R}$, since otherwise it can be too difficult to use $\mathcal{E}$ to discover what the feasible sets are. In the SFM case there is Birkhoff's Representation Theorem [8] that gives a compact representation in terms of a directed graph on $E$. In the BSFM case we can get a similar representation in terms of a directed graph $(\overline{E}, C)$.

### 2.4.1 Skew-Symmetric Representation

Given a directed graph $(E, C)$, we say that $S \subseteq E$ is *closed*, or a (lower) *ideal* if no arc of $C$ exits $S$. Then a representation of an unsigned ring family $\mathcal{S}$ is that it corresponds to the family of ideals of directed graph $(E, C)$ for some $C$. The naive guess would then be that a signed ring family $\mathcal{R}$ should correspond to the family of ideals of directed graph $(\overline{E}, C)$ for some $C$.

One immediate problem arises. Suppose that $C$ contains arc $t \to u$ but not $-u \to -t$. Then the family $\mathcal{R}$ of ideals could contain $S_{tu}$ containing both $t$ and $u$, and also $S_{-u}$, containing $-u$ but neither $t$ nor $-t$. But then $S_{tu} \sqcap S_{-u}$ contains $t$ but not $u$, violating arc $t \to u$ of $C$. Hence to ensure that the family of ideals is a signed ring family, we must assume that $C$ is *skew-symmetric*, i.e., for all arcs $t \to u \in C$, $-u \to -t$ also belongs to $C$.

A second problem is that a closed $S \subseteq \overline{E}$ need not correspond to a valid signed set, since there could be some $u \in S$ such that $-u$ is also in $S$. For $S$ a closed subset of $\overline{E}$, define its *reduced* set $S^0 = S - \bigcup\{\{u, -u\} \mid \text{both } u \text{ and } -u \in S\}$. Then Ando et al. [4, Theorem 3.1] shows that $S^0$ is also closed, and $S^0$ is clearly a signed set. In fact, Reiner [45] and Ando and Fujishige [1] show a signed version of Birkhoff's Representation Theorem that every signed ring family arises as the set of reduced closed sets of a graph $(\overline{E}, C)$ with $C$ skew-symmetric. (Signed ring families can alternatively be defined using closed sets of a *bidirected graph* on node set $E$ [5].) Thus $S \in \mathcal{R}$ means that no arc of $C$ exits $S$ in $(\overline{E}, C)$. We assume from now on that our ring family $\mathcal{R}$ is induced by $(\overline{E}, C)$.

### 2.4.2 Condition Arcs

Note that $t \to u \in C$ then implies the condition that every $S \in \mathcal{R}$ that includes $t$ must also include $u$, and so we call $t \to u \in C$ a *condition arc*. BSFM-SP dynamically adds arcs to $C$ as it progresses, and so $\mathcal{R}$ changes whenever $C$ changes. Its mechanism for finding condition arcs is to find an approximate BSFM solution to the problem BSFM$_t$ where we restrict to sets containing an element $t \in \overline{E}$. If we can prove that every solution to BSFM$_t$ must contain $u$, then it is certainly true that any (unconstrained) BSFM solution that contains $t$ must also contain $u$, and so we add $t \to u$ to $C$. The next lemma shows that when we find condition arcs via this BSFM$_t$ mechanism, then we can force the condition graph $(\overline{E}, C)$ to be skew-symmetric without losing BSFM optimality.

**Lemma 2.1** *If every solution of BSFM$_t$ also contains $u \in \overline{E}$ (so that condition arc $t \to u \in C$), then every solution of BSFM containing $-u$ must also contain $-t$ (so that adding skew condition arc $-u \to -t$ to $C$ preserves optimality).*

**Proof:** Suppose that $S_t$ is an optimal BSFM solution when restricting to sets containing $t$. Since $t \to u \in C$, we know that $u \in S_t$. Therefore if $T$ is any other set containing $t$ but not $u$, then

$$f(S_t) < f(T). \tag{6}$$

Suppose that $R$ solves BSFM, but that $R$ violates condition arc $-u \to -t$, i.e., $R$ contains $-u$ but not $-t$. Now apply (3) to $S_t$ and $R$:

$$f(S_t) + f(R) \ge f(S_t \sqcup R) + f(S_t \sqcap R).$$

Note that since $t \in S_t$ and $-t \notin R$ we have $t \in S_t \sqcup R$, and since $-u \in R$ we have $u \notin S_t \sqcup R$. Now apply (6) to $T = S_t \sqcup R$ to get $f(R) > f(S_t \sqcap R)$, contradicting that $R$ solves BSFM. ■

Note a subtle distinction: The skew-symmetric condition arc $-u \to -t$ we add to $C$ does not mean that every solution to BSFM$_{-u}$ must contain $-t$, only that every BSFM solution containing $-u$ must also contain $-t$, but this is enough for our purposes. Thus we henceforth assume that $C$ is skew-symmetric.

### 2.4.3 Contracting Strong Components

For $u \in \overline{E}$, define $D_u$, the *descendents* of $u$, to be the set of signed elements in $\overline{E}$ reachable from $u$ via directed paths in $(\overline{E}, C)$, and $A_u$, the *ancestors* of $u$, to be the set of signed elements in $\overline{E}$ that can reach $u$ via directed paths in $(\overline{E}, C)$. Note that $t \in D_t$, $A_t$. Skew-symmetry implies that $D_u = -A_{-u}$.

If $(\overline{E}, C)$ has a directed cycle $Q$ and $t \neq u$ are nodes of $Q$, then for any $z \in P_{\mathcal{R}}(f)$ we have $z + \alpha(\text{sgn}(t)\chi^{\text{abs}(t)} - \text{sgn}(u)\chi^{\text{abs}(u)}) \in P_{\mathcal{R}}(f)$ for any (positive or negative) value of $\alpha$, and so $P_{\mathcal{R}}(f)$ cannot have any vertices. Define $\mathcal{S}_{\text{all}}$ to be the family of node sets of strongly connected components of $(\overline{E}, C)$, so that each $\tau \in \mathcal{S}_{\text{all}}$ is a subset of $\overline{E}$. The signed elements in such a $\tau$ must either all belong to a BSFM solution, or none of them belongs, and so w.l.o.g. we contract the strong components so that the polyhedron on the contracted problem does have vertices that we can work with.

Suppose that for some $\tau \in \mathcal{S}_{\text{all}}$ there is some $e \in E$ with $+e$, $-e \in \tau$, and suppose that $t \in \overline{E}$ is some other element of $\tau$. Then by strong connectivity of $\tau$ there are paths from $+e$ to $t$ and from $t$ to $+e$. By skew-symmetry then there are also paths from $-t$ to $-e$ and from $-e$ to $-t$, and so $-t$ also belongs to $\tau$. This shows that such a $\tau$ is self-skew, and that no element of such a $\tau$ can belong to an optimal BSFM solution. We define $\mathcal{S}_{\text{self}}$ to be the set of such $\tau$, and define $\mathcal{S}_{\text{cur}} = \mathcal{S}_{\text{all}} - \mathcal{S}_{\text{self}}$.

Note that for $\tau \in \mathcal{S}_{\text{cur}}$ we have that $-\tau \cap \tau = \emptyset$, and by skew-symmetry $-\tau$ is also a strong component. Hence strong components in $\mathcal{S}_{\text{cur}}$ come in skew-symmetric pairs; for each skew-symmetric pair $\tau$, $-\tau \in \mathcal{S}_{\text{cur}}$, arbitrarily designate one as positive and the other as negative. Define $\mathcal{S}^+_{\text{cur}}$ as the set of positive elements of $\mathcal{S}_{\text{cur}}$, and $\mathcal{S}^-_{\text{cur}}$ as the set of negative elements.

We usually work with the members of $\mathcal{S}_{\text{cur}}$ as our "elements", which we also call "nodes". At the end we need to recover a solution in terms of the signed elements in $\overline{E}$. Note that $\mathcal{S}_{\text{cur}}$ gets recursively contracted as more arcs are added to $C$, and so we need to recursively expand it at the end. We use $\text{abs}(\mathcal{S}_{\text{cur}})$ to denote the family of unsigned strong components, which has just one member for each signed pair in $\mathcal{S}_{\text{cur}}$. Define $\mathcal{R}_{\text{cur}}$ to be the current family of reduced closed sets of $(\mathcal{S}_{\text{cur}}, C)$, i.e., the signed ring family of feasible sets for $\mathcal{S}_{\text{cur}}$. For $S = (S^+, S^-) \in \mathcal{R}_{\text{cur}}$ define $\overline{E}(S)$ to be $\bigcup_{\tau \in S^+} \tau \cup \bigcup_{\tau \in S^-} -\tau$, so that $S \in \mathcal{R}_{\text{cur}}$ iff $\overline{E}(S) \in \mathcal{R}$.

Now define $\hat{f} : \mathcal{R}_{\text{cur}} \to \mathbb{R}$ by $\hat{f}(S) = f(\overline{E}(S))$. It is easy to check that $\hat{f}$ is bisubmodular, and that if $Z$ solves BSFM for $\hat{f}$, then $\overline{E}(Z)$ solves BSFM for $f$. Hence we reduce BSFM for $f$ to the smaller problem of BSFM for $\hat{f}$, with polyhedron $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$. We consider the condition arcs in $C$ whose ends are in different components of $\mathcal{S}_{\text{cur}}$ as being arcs on $\mathcal{S}_{\text{cur}}$ as well as arcs on $\overline{E}$. Note that the graph $(\mathcal{S}_{\text{cur}}, C)$ is acyclic.

### 2.4.4 Partitioning the Skew Graph

Note that for $t \in \overline{E}$, $D_t$ and $A_t$ are unions of elements of $\mathcal{S}_{\text{cur}}$, so it makes sense to define $D_\tau$ and $A_\tau$ via reachability from and to $\tau$ in the graph $(\mathcal{S}_{\text{cur}}, C)$. For some $\tau \in \mathcal{S}_{\text{cur}}$, $C$ might include the arc $\tau \to -\tau$ (this arc is self-skew), which implies that no $S \in \mathcal{R}_{\text{cur}}$ can include $\tau$. Since conditions in $C$ are transitive, for such a $\tau$ if $\rho \in A_\tau$ then also $-\rho \in D_{-\tau}$, giving a path in $C$ from $\rho$ to $-\rho$, showing that no $S \in \mathcal{R}_{\text{cur}}$ can include $\rho$. We collect components $\tau \in \mathcal{S}_{\text{cur}}$ such that $-\tau \in D_\tau$ into $\mathcal{S}_{\text{out}}$, the components known to be *out* of every BSFM solution for $\hat{f}$. We can and do assume w.l.o.g. (by adding these arcs to $C$ if necessary) that $\tau \in \mathcal{S}_{\text{out}}$ iff $\tau \to -\tau \in C$. Note that $\mathcal{S}_{\text{out}} \subseteq \mathcal{S}_{\text{cur}}$, and that no arc of $C$ enters $\mathcal{S}_{\text{out}}$.

The only components that appear in signed sets in $\mathcal{R}_{\text{cur}}$ are the *active* strong components $\mathcal{S}_{\text{act}} = \mathcal{S}_{\text{cur}} - \mathcal{S}_{\text{out}}$. In fact $\mathcal{R}_{\text{cur}}$ is exactly the family of closed sets in $\mathcal{S}_{\text{act}}$ (and this partition of $\mathcal{S}_{\text{all}}$ into $\mathcal{S}_{\text{self}}$, $\mathcal{S}_{\text{out}}$, and $\mathcal{S}_{\text{act}}$ gives the essential idea of Ando and Fujishige's [1] proof of the signed version of Birkhoff's Theorem). Also define index sets of unsigned components $\mathcal{S}^P_{\text{act}} = \{\eta \in \text{abs}(\mathcal{S}_{\text{cur}}) \mid +\eta \in \mathcal{S}_{\text{act}}\}$ and $\mathcal{S}^N_{\text{act}} = \{\eta \in \text{abs}(\mathcal{S}_{\text{cur}}) \mid -\eta \in \mathcal{S}_{\text{act}}\}$.

For simplicity we henceforth denote elements of $\mathcal{S}_{\text{cur}}$ with letters $u$, $t$, $s$, $\dots$, and elements of $\text{abs}(\mathcal{S}_{\text{cur}})$ with letters $e$, $g$, $\dots$, despite using the same letters for the elements which constitute the members of $\mathcal{S}_{\text{cur}}$ and $\text{abs}(\mathcal{S}_{\text{cur}})$. We use the same capital letters to distinguish sets of signed elements from sets of unsigned elements, except that we keep $D_t$ and $A_t$ as signed sets.

Therefore at a generic step of the algorithm we have $\mathcal{S}_{\text{all}}$ as the strong components of $(\overline{E}, C)$, partitioned into the self-skew set $\mathcal{S}_{\text{self}}$, and the current working set of disjoint skew pairs $\mathcal{S}_{\text{cur}}$. Then $\mathcal{S}_{\text{cur}}$ is in turn partitioned into $\mathcal{S}_{\text{out}}$ and $\mathcal{S}_{\text{act}}$, where for $u \in \mathcal{S}_{\text{cur}}$ we could have either that both $u$ and $-u$ are in $\mathcal{S}_{\text{act}}$, or only one of them is. The family $\mathcal{R}_{\text{cur}}$ of reduced closed sets of $(\mathcal{S}_{\text{cur}}, C)$ is a signed ring family, and $\hat{f} : \mathcal{R}_{\text{cur}} \to \mathbb{R}$ is a bisubmodular function defined on $\mathcal{R}_{\text{cur}}$. Figure 1 shows a picture of this partition.

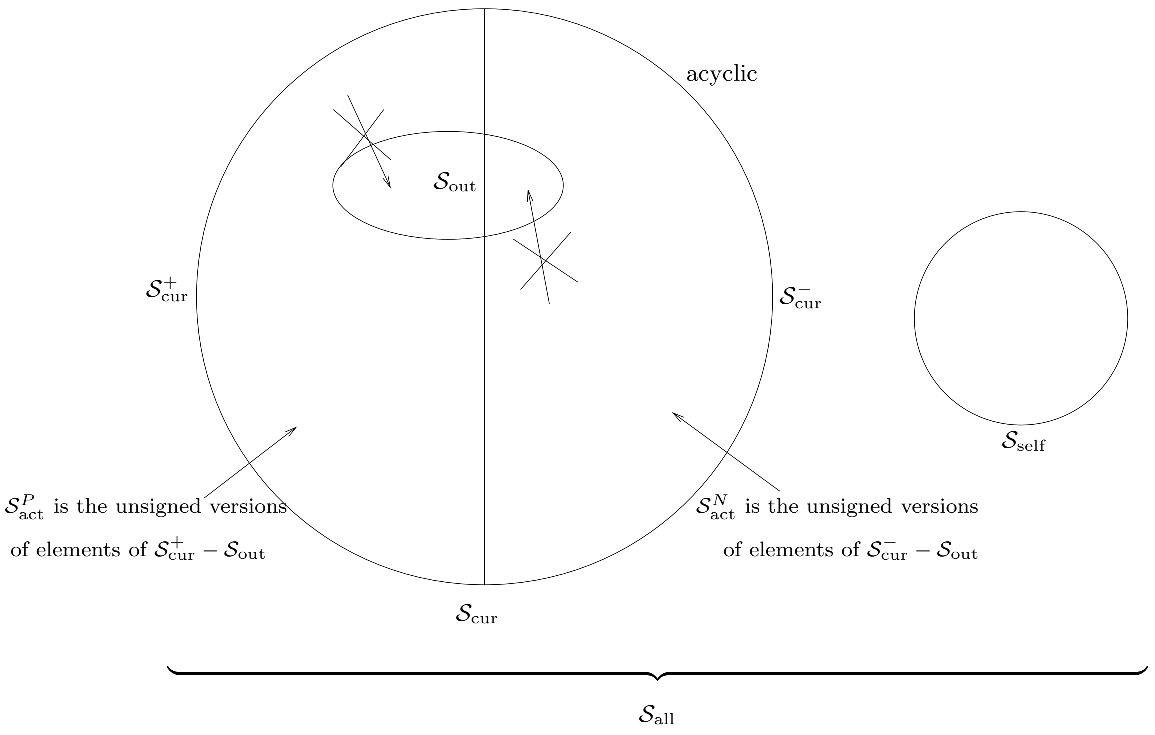
acyclic
$\mathcal{S}_{\text{out}}$
$\mathcal{S}^+_{\text{cur}}$
$\mathcal{S}^-_{\text{cur}}$
$\mathcal{S}_{\text{self}}$
$\mathcal{S}^P_{\text{act}}$ is the unsigned versions of elements of $\mathcal{S}^+_{\text{cur}} - \mathcal{S}_{\text{out}}$
$\mathcal{S}^N_{\text{act}}$ is the unsigned versions of elements of $\mathcal{S}^-_{\text{cur}} - \mathcal{S}_{\text{out}}$
$\mathcal{S}_{\text{cur}}$
$\mathcal{S}_{\text{all}}$

Figure 1: Picture of the structure of the partition of the contracted skew sets $\mathcal{S}_{\text{all}}$ into $\mathcal{S}_{\text{cur}}$ and $\mathcal{S}_{\text{self}}$, and then $\mathcal{S}_{\text{cur}}$ into $\mathcal{S}_{\text{act}}$ and $\mathcal{S}_{\text{out}}$.

When BSFM-SP dynamically adds arcs to $C$ we call subroutine UPDATES to re-compute $\mathcal{S}_{\text{all}}$, $\mathcal{S}_{\text{self}}$, $\mathcal{S}_{\text{out}}$, and $\mathcal{S}_{\text{act}}$. For example if $u \in \mathcal{S}_{\text{out}}$ (i.e., $u \to -u \in C$ and $-u \in \mathcal{S}_{\text{act}}$) and new arcs of $C$ create a path from $-u$ to $u$, then the components $u$ and $-u$ of the old $\mathcal{S}_{\text{all}}$ (possibly plus other components included on these paths) collapse into a new larger component that moves to $\mathcal{S}_{\text{self}}$. The time for UPDATES is $O(n|C|) \leq O(n^3)$, which is dominated by other operations.

## 2.5 Signed Greedy Optimizes over Bisubmodular Polyhedra

A *sign vector* (on $\text{abs}(\mathcal{S}_{\text{cur}})$) is some $\sigma \in \{+,-\}^{\text{abs}(\mathcal{S}_{\text{cur}})}$. Sign vector $\sigma$ partitions $\text{abs}(\mathcal{S}_{\text{cur}})$ into $\text{abs}(\mathcal{S}_{\text{cur}})^+(\sigma) = \{e \in \text{abs}(\mathcal{S}_{\text{cur}}) \mid \sigma_e = +\}$ and $\text{abs}(\mathcal{S}_{\text{cur}})^-(\sigma) = \{e \in \text{abs}(\mathcal{S}_{\text{cur}}) \mid \sigma_e = -\}$. If $S \subseteq \text{abs}(\mathcal{S}_{\text{cur}})$ is an unsigned subset, its signed counterpart w.r.t. $\sigma$, denoted $S|\sigma$, is the signed set $(S \cap \text{abs}(\mathcal{S}_{\text{cur}})^+(\sigma), S \cap \text{abs}(\mathcal{S}_{\text{cur}})^-(\sigma))$.

Suppose that we have a pair of a linear order $\prec$ and sign vector $\sigma$ on $\text{abs}(\mathcal{S}_{\text{cur}})$. It is convenient to let $\sigma$ define a *signed linear order* $\prec_\sigma$ such that if $t = \sigma_e e$, $u = \sigma_g g$, and $e \prec g$, then we say that $t \prec_\sigma u$. It is further useful to let $\prec_\sigma$ also stand for $\text{abs}(\mathcal{S}_{\text{cur}})|\sigma$ and so write $t \in \prec_\sigma$ to mean that $t = \sigma_{\text{abs}(t)}\text{abs}(t)$. We call the pair $\prec, \sigma$ *consistent* (with $(\mathcal{S}_{\text{cur}}, C)$) if for every $t \in \prec_\sigma$ we have $t \in \mathcal{S}_{\text{act}}$ (so that $\prec_\sigma \subseteq \mathcal{S}_{\text{act}}$); and for every $t \to u \in C$, if $t \in \prec_\sigma$ then $u \in \prec_\sigma$ and $u \prec_\sigma t$.

For a consistent pair $\prec$, $\sigma$, and any $e \in \text{abs}(\mathcal{S}_{\text{cur}})$, define $e^\prec$ as $\{g \in \text{abs}(\mathcal{S}_{\text{cur}}) \mid g \prec e\}$, a subset of $\text{abs}(\mathcal{S}_{\text{cur}})$, and define $e^\prec_{n+1} = \text{abs}(\mathcal{S}_{\text{cur}})$. Note that $\prec$, $\sigma$ consistent implies that for every $e \in \text{abs}(\mathcal{S}_{\text{cur}})$, $e^\prec|\sigma$ is a reduced closed set of $(\mathcal{S}_{\text{cur}}, C)$ (and so contains $D_{\sigma_e e} - (\sigma_e e)$) for every $e \in \text{abs}(\mathcal{S}_{\text{cur}})$, i.e., that $e^\prec|\sigma \in \mathcal{R}_{\text{cur}}$. Note that since $(\mathcal{S}_{\text{cur}}, C)$ is acyclic, a consistent pair $\prec, \sigma$ always exists.

The Greedy Algorithm generates vertices of the submodular polyhedron, and so it is natural to generalize it to the *Signed Greedy Algorithm*. Signed Greedy takes consistent $\prec$ and $\sigma$ as input, and outputs a vector $v^{\prec,\sigma} \in \mathbb{R}^{\text{abs}(\mathcal{S}_{\text{cur}})}$; component $e_i$ of $v^{\prec,\sigma}$ is then $v^{\prec,\sigma}_{e_i}$.

**The Signed Greedy Algorithm**

```
    Input: Consistent linear order $\prec = (e_1, e_2, \ldots, e_n)$ and sign vector $\sigma$.
    Output: Vertex $v^{\prec,\sigma}$.

    For $i = 1, \ldots, n$
        Set $v^{\prec,\sigma}_{e_i} = \sigma_{e_i}(\hat{f}(e^\prec_{i+1}|\sigma) - \hat{f}(e^\prec_i|\sigma))$ ( $= \sigma_{e_i}[\hat{f}((e^\prec_i + e_i)|\sigma) - \hat{f}(e^\prec_i|\sigma)]$).
    Return $v^{\prec,\sigma}$.
```

It is known (see [2, 11, 17, 35, 41]) that the output $v^{\prec,\sigma}$ of Signed Greedy is a vertex of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, and conversely for every vertex $v$ of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, there is a consistent pair $\prec, \sigma$ such that Signed Greedy applied to $\prec, \sigma$ produces $v$. Our ability to use Signed Greedy to generate vertices of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ is a key part of our BSFM algorithms.

Suppose that $y \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$. We say that $S \in \mathcal{R}_{\text{cur}}$ is *tight* for $y$ if $y(S) = \hat{f}(S)$. A corollary to Signed Greedy is that

If $v^{\prec,\sigma}$ is generated by Signed Greedy from $\prec, \sigma$, then $e^\prec|\sigma$ is tight for $v^{\prec,\sigma}$ for all $e \in \text{abs}(\mathcal{S}_{\text{cur}})$. (7)

## 2.6 BSFM Optimality on Signed Ring Families

For a scalar $\mu \in \mathbb{R}$ define $\mu^+ = \max(\mu, 0)$ and $\mu^- = \min(\mu, 0)$ as the positive and negative parts of $\mu$. For $H \subseteq \text{abs}(\mathcal{S}_{\text{cur}})$ this extends to vectors $w \in \mathbb{R}^{\text{abs}(\mathcal{S}_{\text{cur}})}$ via $w^+(H) = \sum_{e \in H} w^+_e$ and similarly for $w^-(H)$. Define $||w||_{\text{cur}} = w^+(\mathcal{S}^N_{\text{act}}) - w^-(\mathcal{S}^P_{\text{act}})$. Note that if $\mathcal{S}_{\text{out}} = \emptyset$, then $\mathcal{S}_{\text{act}} = \mathcal{S}_{\text{cur}}$ so that $\mathcal{S}^P_{\text{act}} = \mathcal{S}^N_{\text{act}} = \text{abs}(\mathcal{S}_{\text{cur}})$, and then $||w||_{\text{cur}}$ is the $\ell_1$ norm $||w||_1$ of $w$.

**Theorem 2.2 ([20, Theorem 3.1])** *For any bisubmodular $\hat{f} : \mathcal{R}_{\text{cur}} \to \mathbb{R}$,*

$$\min_{S \in \mathcal{R}_{\text{cur}}} \hat{f}(S) = \max_{w \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})} -||w||_{\text{cur}}. \tag{8}$$

*Moreover, if $\hat{f}$ is integer-valued, then the maximum on the right-hand side of (8) is attained by an integral $w$.* ∎

If $w \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ and $S \in \mathcal{R}_{\text{cur}}$ are any feasible solutions, then

$$\begin{array}{rcl} -||w||_{\text{cur}} & = & w^-(\mathcal{S}^P_{\text{act}}) - w^+(\mathcal{S}^N_{\text{act}}) \\ & \leq & w^-(S^+) - w^+(S^-) \\ & \leq & w(S^+) - w(S^-) \\ & \leq & \hat{f}(S), \end{array} \tag{9}$$

which is weak duality for (8). Optimality for $w$ and $S$ is equivalent to complementary slackness, which is the condition that all the inequalities in (9) become equalities, or

(CS i) if $w_e > 0$ then $e \in S^-$, and if $w_e < 0$ then $e \in S^+$ (the first inequality is tight); this implies that $w_e \leq 0$ for $e \in S^+$, and $w_e \geq 0$ for $e \in S^-$ (the second inequality is tight); and

(CS ii) $S$ is tight for $w$, i.e., $w(S^+) - w(S^-) = \hat{f}(S)$ (the third inequality is tight).

## 2.7 Representing Solutions

We need some further machinery to represent points $w \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, and to get a handle on the size of solutions.

### 2.7.1 Flow Boundaries in Skew Graphs

If $\varphi$ is a flow in a graph $(\mathcal{S}_{\text{cur}}, C)$, then for $t \in \mathcal{S}_{\text{cur}}$ we define the *boundary* of $\varphi$ at $t$ (net flow out of $t$) as $\partial\varphi_t = \sum_{t \to u \in C} \varphi_{tu} - \sum_{r \to t \in A} \varphi_{rt}$. Since boundary is modular, the net flow out of $S \subseteq \mathcal{S}_{\text{cur}}$ is $\partial\varphi(S) = \sum_{t \in S} \partial\varphi_t$. Flow $\varphi$ on a skew-symmetric graph such as $(\mathcal{S}_{\text{cur}}, C)$ is called a *skew flow* if $\varphi_{tu} = \varphi_{-t,-u}$ for all $t \to u \in C$. A skew flow satisfies $\partial\varphi_t = -\partial\varphi_{-t}$. For $e \in \text{abs}(\mathcal{S}_{\text{cur}})$ define $\partial^+\varphi_e = \partial\varphi_{+e}$. Then $\varphi$ skew implies that $\partial\varphi_{-e} = -\partial\varphi_{+e} = -\partial^+\varphi_e$, and so for $S = (S^+, S^-)$, $\partial\varphi(S) = \partial^+\varphi(S^+) - \partial^+\varphi(S^-)$.

Suppose that $\varphi$ is a skew flow on $C$ satisfying $\varphi \geq 0$. If $S \in \mathcal{R}_{\text{cur}}$ then no arc of $C$ exits $S$, and so $\partial\varphi(S) \leq 0$. Suppose that $y \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ so that $y(S) \leq \hat{f}(S)$ for all $S \in \mathcal{R}_{\text{cur}}$, and define $w = y + \partial^+\varphi$. Then $w(S) = (y + \partial^+\varphi)(S^+) - (y + \partial^+\varphi)(S^-) = y(S) + \partial\varphi(S) \leq y(S) \leq \hat{f}(S)$, and so $w$ also belongs to $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$. Since we could multiply $\varphi$ by any positive scalar, we see that $\partial^+\varphi$ is a direction of unboundedness of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$. Ando and Fujishige [2] showed that the converse is also true: Every point $w \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ can be represented as $w = y + \partial^+\varphi$, where $y$ is a convex combination of the vertices of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, and $\varphi$ is a non-negative skew flow on $C$.

### 2.7.2 Convex Combination of Vertices

As in other combinatorial SFM and BSFM algorithms it is non-trivial to verify that a point $w$ belongs to $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, since it could have an exponential number of constraints. As in the previous section we keep the unbounded part of $w$ as the boundary of skew flow $\varphi \geq 0$ on $C$, and we keep the bounded part $y$ using the idea originated by Cunningham [13]: We keep an index set $\mathcal{I}$ of vertices $v^i$ of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, where $v^i = v^{\prec^i, \sigma^i}$ is generated from Signed Greedy applied to consistent pair $\prec^i, \sigma^i$. We also keep scalar multipliers $\lambda_i \geq 0$ so that our current point $y \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ is represented as a convex combination of vertices of $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ via

$$\sum_{i \in \mathcal{I}} \lambda_i = 1, \quad y = \sum_{i \in \mathcal{I}} \lambda_i v^i. \tag{10}$$

Thus we keep the invariant that our current point $w \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ equals $y + \partial^+\varphi$. Whenever we modify representation (10) we need to ensure that any new vertices come from consistent pairs (also needed for Lemma 3.11).

Because $\prec^i$ is always associated with $\sigma^i$, we simplify and abuse notation and let $\prec^i$ stand for both the unsigned order on $\text{abs}(\mathcal{S}_{\text{cur}})$ and the signed order $\prec^i_{\sigma^i}$ (though we sometimes continue to use $\prec^i_{\sigma^i}$ when we want to emphasize that we are referring to the signed version). Since we were already using $\prec^i_{\sigma^i}$ to stand for $\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i$, we further overload $\prec^i$ to also stand for $\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i$. Therefore writing $t \in \prec^i$ means that $\sigma^i_{\text{abs}(t)} = \text{sgn}(t)$.

Suppose that we have a signed set $S \in \mathcal{R}_{\text{cur}}$ and a point $w = y + \partial^+\varphi$ with $y$ represented as in (10) as a guess at BSFM solutions. Define the set $W^w(S)$ of elements *wrong for* $w$ by $e \in W^w(S)$ if $+e \in S$ and $w_e > 0$ or $-e \in S$ and $w_e < 0$ (wrong elements violate (CS i)). For each $i \in \mathcal{I}$, define $G^i(S) = \text{abs}(S \cap \prec^i_{\sigma^i})$, the $e \in \text{abs}(\mathcal{S}_{\text{cur}})$ with $\sigma^i_e e \in S$ (good elements of $i$ belong to $S$ with the correct sign); and $B^i(S) = \text{abs}((-S) \cap \prec^i_{\sigma^i})$ as the $e \in \text{abs}(\mathcal{S}_{\text{cur}})$ with $-\sigma^i_e e \in S$ (bad elements of $i$ belong to $S$ with the wrong sign).

Since $S \in \mathcal{R}_{\text{cur}}$, no arc of $C$ exits $S$. Suppose that $\varphi_{st} = 0$ for every $s \to t \in C$ entering $S$ so that $\partial^+\varphi(S) = 0$, implying that $w(S) = y(S)$. Suppose further that for every $i \in \mathcal{I}$, $B^i(S) = \emptyset$ so that $G^i(S) = \text{abs}(S)$, and that the elements of $\text{abs}(S)$ are left-most in $\prec^i$. This is equivalent to saying that for every $i \in \mathcal{I}$, there is some $e \notin \text{abs}(S)$ such that $S = e^{\prec^i}|\sigma^i$, and so by (7) that $S$ is tight for $v^i$. Therefore by (10), $y(S) = \sum_{i \in \mathcal{I}} \lambda_i v^i(S) = \sum_{i \in \mathcal{I}} \lambda_i \hat{f}(S) = \hat{f}(S)$, and so $S$ is also tight for $w$. If in addition $W^w(S) = \emptyset$, then (CS i) is satisfied, and $S$ tight for $w$ verifies that (CS ii) is satisfied, proving optimality of $S$ and $w$.

This gives part of the idea of our algorithms: Our current $w$ and $S$ define wrong elements for $w$, and good and bad elements for each $i$. We try to modify $w$ by modifying representation (10) so as to drive $w_e$ for $e \in W^w(S)$ towards zero, and for each $i \in \mathcal{I}$, elements of $G^i(S)$ to the left and elements of $B^i(S)$ to the right. When $e \in B^i(S)$ becomes the right-most in $\prec^i$, then we can flip its sign $\sigma^i_e$ to move $e$ into $G^i(S)$, at which point we then try to drive $e$ back to the left. It appears to be necessary to drive $e \in B^i(S)$ to the extreme right of $\prec^i$ before flipping their signs because it is difficult to manage the changes in the components of $v^i$ if we change signs in the middle of $\prec^i$. We also try to drive the flow $\varphi$ into $S$ to zero.

### 2.7.3 Size of $\hat{f}$

We need an estimate of the "size" of $\hat{f}$ sharper than $M$ for the analyses of BSFM-SP and BSFM-FC. Compute $\delta^0 = \max_{t \in \mathcal{S}_{\text{act}}} \hat{f}(D_t) - \hat{f}(D_t - t)$. Intuitively, (4) and Signed Greedy say that the magnitude of $v_e^{\prec,\sigma}$ is larger as $e$ occurs sooner in $\prec$. Consistency says that $D_{\sigma_e e} - (\sigma_e e)$ must precede $e$ in $\prec$, so the largest magnitude for $v_e^{\prec,\sigma}$ should occur when $D_{\sigma_e e}$ occurs first in $\prec$ immediately followed by $e$. Then $\sigma_e v_e^{\prec,\sigma} = \hat{f}(D_{\sigma_e e}) - \hat{f}(D_{\sigma_e e} - (\sigma_e e)) \le \delta^0$, and so the size of all components should be at most $\delta^0$. The next lemma formalizes this.

**Lemma 2.3** *Suppose that $y$ belongs to the convex hull of vertices $v^{\prec,\sigma}$ of $P_{\mathcal{R}_{\rm cur}}(\hat{f})$. If $e \in \mathcal{S}^P_{\rm act}$ then $y_e \le +\delta^0$, and if $e \in \mathcal{S}^N_{\rm act}$ then $y_e \ge -\delta^0$.*

**Proof:** It suffices to prove the statement for $y = v^{\prec,\sigma}$ for some $\prec, \sigma$ consistent with $(\mathcal{S}_{\rm cur}, C)$. Signed Greedy computes $\sigma_e y_e = \hat{f}((e^{\prec}+e)|\sigma) - \hat{f}(e^{\prec}|\sigma)$. Consistency implies that $D_{\sigma_e e} \sqsubseteq (e^{\prec}+e)|\sigma$, and then (4) implies that $\sigma_e y_e = \hat{f}((e^{\prec}+e)|\sigma) - \hat{f}(e^{\prec}|\sigma) \le \hat{f}(D_{\sigma_e e}) - \hat{f}(D_{\sigma_e e} - (\sigma_e e)) \le \delta^0$. Thus if $\sigma_e = +$ we have $y_e \le \delta^0$, and if $\sigma_e = -$ we have $y_e \ge -\delta^0$.

Now suppose that $\sigma_e = +$, so that $e \in \mathcal{S}^P_{\rm act}$, and that $e$ also belongs to $\mathcal{S}^N_{\rm act}$. Then $y_e = \hat{f}((e^{\prec}+e)|\sigma) - \hat{f}(e^{\prec}|\sigma)$, and we must show that $y_e \ge -\delta^0$. We first claim that there exists a consistent pair $\prec'$, $\sigma'$ such that $e$ is right-most in $\prec'$, $\sigma'_e = \sigma_e$, $(e^{\prec}+e)|\sigma \sqsubseteq (e^{\prec'}+e)|\sigma'$, and if we define $\sigma''$ to be $\sigma'$ except that $\sigma''_e = -\sigma'_e$ then $\prec'$, $\sigma''$ is also consistent. Suppose that there is some $g \in \text{abs}(\mathcal{S}_{\rm cur})$ with $g \in e^{\prec}$, $\sigma_g = +$ (the case with $\sigma_g = -$ is similar) and $-g \in D_{-e}$; this would be bad as it would require that $\sigma'_g = \sigma_g$ to ensure that $(e^{\prec}+e)|\sigma \sqsubseteq (e^{\prec'}+e)|\sigma'$, but $\sigma'_g = -\sigma_g$ to make $\prec'$, $\sigma''$ consistent. But by skewness of $C$, $-g \in D_{-e}$ implies that $+e \in D_{+g}$, and then consistency of $\prec, \sigma$ would imply that $e \prec g$, contradicting that $g \prec e$. Therefore there is a way to choose a $\prec'$ and the signs of $\sigma'$ such that $\prec'$ is consistent with both $\sigma'$ and $\sigma''$, and such that $(e^{\prec}+e)|\sigma \sqsubseteq (e^{\prec'}+e)|\sigma'$.

Since $-e \in \prec'_{\sigma''}$, we have that $D_{-e} \sqsubseteq \text{abs}(\mathcal{S}_{\rm cur})|\sigma''$. Then we get that

$$\begin{array}{rcll}
y_e & = & \hat{f}((e^{\prec}+e)|\sigma) - \hat{f}(e^{\prec}|\sigma) = v_e^{\prec,\sigma} & \text{definition of } y_e \\
& \ge & \hat{f}((e^{\prec'}+e)|\sigma') - \hat{f}(e^{\prec'}|\sigma') = v_e^{\prec',\sigma'} & \text{(4) and } (e^{\prec}+e)|\sigma \sqsubseteq (e^{\prec'}+e)|\sigma' \\
& = & \hat{f}(\text{abs}(\mathcal{S}_{\rm cur})|\sigma') - \hat{f}((\text{abs}(\mathcal{S}_{\rm cur}) - e)|\sigma') & e \text{ right-most in } \prec' \\
& \ge & \hat{f}((\text{abs}(\mathcal{S}_{\rm cur}) - e)|\sigma') - \hat{f}(\text{abs}(\mathcal{S}_{\rm cur})|\sigma'') & (5) \\
& = & -[\hat{f}(\text{abs}(\mathcal{S}_{\rm cur})|\sigma'') - \hat{f}((\text{abs}(\mathcal{S}_{\rm cur}) - e)|\sigma'')] = v_e^{\prec',\sigma''} & \sigma' = \sigma'' \text{ outside } e \\
& \ge & -[\hat{f}(D_{-e}) - \hat{f}(D_{-e} - (-e))] & \text{(4) and } D_{-e} \sqsubseteq \text{abs}(\mathcal{S}_{\rm cur})|\sigma'' \\
& \ge & -\delta^0 & \text{definition of } \delta^0
\end{array}$$

A similar argument shows that if $\sigma_e = -$ and $e \in \mathcal{S}^P_{\rm act}$, then $y_e \le \delta^0$. ■

## 3 The New BSFM Algorithms

Iwata, Fleischer, and Fujishige's SFM algorithm [33] uses the successive approximation framework of Goldberg and Tarjan [26] and Tardos [50], where the complementary slackness conditions for the original problem are relaxed by a scaling parameter $\delta$. Roughly speaking, we call a solution *$\delta$-optimal* if it satisfies these $\delta$-approximate conditions. Then a Refine subroutine is developed that takes a $\delta$-optimal solution, cuts $\delta$ in half, and then re-optimizes to get a new solution that is $\delta$-optimal for the new $\delta$. Each call to Refine is also called a *scaling phase*.

The IFF Refine subroutine runs in $O(n^5\text{EO})$ time. Then IFF [33] embeds this into $O(\log M)$ scaling phases to get its weakly polynomial $O(n^5\text{EO}\log M)$ running time, and into ($O(n^2)$ calls to a fixing routine) times ($O(\log n)$ scaling phases per fix) to get its $O(n^7 \log n\text{EO})$ strongly polynomial running time. Iwata [30] modifies Refine to make it fully combinatorial at the cost of making it a factor of $O(n^2 \log n)$ slower. Otherwise its running time analysis is similar to IFF-SP's, for its $O(n^9 \log^2 n\mathrm{EO})$ running time.

Turning now to BSFM, Fujishige and Iwata [24] use the same successive approximation framework as IFF. The refine subroutine in FuI-BSFM, which we call BRefine, runs in the same $O(n^5\mathrm{EO})$ time as Refine. We extend BRefine to subroutine BRefine$\mathcal{R}$ that works directly over signed ring families, also in $O(n^5\mathrm{EO})$ time. BRefine$\mathcal{R}$ again takes a $\delta$-optimal solution, cuts $\delta$ in half, and then re-optimizes to get a new solution that is $\delta$-optimal for the new $\delta$, and one invocation is again called a *scaling phase*. Since our BRefine$\mathcal{R}$ is quite similar to BRefine, we mostly emphasize only where differences occur due to the need to handle the signed ring family.

We first show the basic operations used by the algorithm. Then we show how to imbed the basic algorithm into a strongly polynomial framework different from IFF (first developed in [36]) to get a strongly polynomial algorithm. We then can use this strongly polynomial algorithm in much the same way as Iwata [30] to get a fully combinatorial algorithm.

## 3.1 Basic Framework of the Algorithms

We have a bisubmodular function $\hat{f}$ over the set of elements $\mathrm{abs}(\mathcal{S}_{\mathrm{cur}})$ with the signed ring family $\mathcal{R}_{\mathrm{cur}}$ as the family of feasible sets, represented as the reduced closed sets of skew-symmetric acyclic graph $(\mathcal{S}_{\mathrm{cur}}, C)$. Recall that $\mathcal{S}_{\mathrm{cur}}$ is a superset of $\mathcal{S}_{\mathrm{act}}$ such that if $t \in \mathcal{S}_{\mathrm{act}}$, then $-t \in \mathcal{S}_{\mathrm{cur}}$ (whereas $-t$ might not be in $\mathcal{S}_{\mathrm{act}}$, if $-t \in \mathcal{S}_{\mathrm{out}}$). Unlike most other papers in this area, we specify our algorithm to be able to directly handle the signed ring family $\mathcal{R}_{\mathrm{cur}}$ from the start. Although this makes the algorithm somewhat more complicated, it has two advantages: (1) It shows that solving BSFM over a signed ring family is essentially no harder than over $3^E$, and (2) Lemma 3.11 below needs this capability to make the fully combinatorial version work correctly.

We keep a current point $w \in P_{\mathcal{R}_{\mathrm{cur}}}(\hat{f})$, represented as $w = y + \partial^+\varphi$, where $\varphi \geq 0$ is a skew flow on $C$, and $y$ is represented as in (10). We keep a second skew flow $x$ on the complete directed graph on $\mathcal{S}_{\mathrm{cur}}$ (specifically including arcs $t \to -t$ for all $t \in \mathcal{S}_{\mathrm{cur}}$), namely $(\mathcal{S}_{\mathrm{cur}}, A_{\mathrm{relax}})$, where $A_{\mathrm{relax}}$ are the *relaxation* arcs. For all $t \to u \in A_{\mathrm{relax}}$ we enforce that $x_{tu} \cdot x_{ut} = 0$. We put bounds $0 \leq x_{tu} \leq \delta$ for all $t \to u \in A_{\mathrm{relax}}$, and $0 \leq \varphi_{tu} < \infty$ for $t \to u \in C$ (the infinite bounds make BRefine$\mathcal{R}$ consider only sets in $\mathcal{R}_{\mathrm{cur}}$, see below). We call $x$ satisfying these bounds *$\delta$-feasible.* (This flow relaxation is expressed differently from [24], but it amounts to the same idea.) Define

$$z = w + \partial^+ x = y + \partial^+\varphi + \partial^+ x. \tag{11}$$

Instead of concentrating on minimizing $||w||_{\mathrm{cur}}$, BRefine$\mathcal{R}$ concentrates on minimizing $||z||_{\mathrm{cur}}$; as $\delta$ decreases, $w$ and $z$ converge together. It does this through two operations:

1. Try to Augment flow $x$ on paths (without changing $y$, but possibly changing $\varphi$ and hence $w$ to keep (11) invariant);

2. When blocked from augmenting, it modifies $y$ by (fully or partially) replacing one vertex in (10) with another vertex (and modifies $x$ to keep (11) invariant).

### 3.1.1 The Augmentation Network

We want each augmentation to be by amount $\delta$, so we define the set of *residual* arcs where flow can be changed by $\delta$ as $R(x) = \{t \to u \in A_{\mathrm{relax}} \mid x_{tu} = 0\}$ (if $x_{ut} = \delta$ so that we could decrease

$x$ by $\delta$ on $u \to t$ and would want to include $t \to u$ in $R(x)$, due to enforcing $x_{tu} \cdot x_{ut} = 0$ we would have $x_{tu} = 0$ and so $t \to u$ would be in $R(x)$ anyway). Note that since $x$ (and the bounds) are skew-symmetric, $R(x)$ is also skew-symmetric, so that for any $t, u \in \mathcal{S}_{\text{cur}}$, there is a directed path $P$ using arcs of $R(x)$ from $t$ to $u$ iff $-P$ (the skew version of $P$) is a directed path using arcs of $R(x)$ from $-u$ to $-t$ (however $P$ and $-P$ need not be arc-disjoint).

The target set of unsigned nodes where we want to decrease $z$ by $\delta$ is $UT^{+\delta}(z) = \{l \in \mathcal{S}^N_{\text{act}} \mid z_l \geq +\delta\}$, and where we want to increase $z$ by $\delta$ is $UT^{-\delta}(z) = \{l \in \mathcal{S}^P_{\text{act}} \mid z_l \leq -\delta\}$, as these would decrease $||z||_{\text{cur}}$. Define the target signed source set $T_{\text{source}}(z) = (UT^{-\delta}(z), UT^{+\delta}(z))$, and the target signed sink set $T_{\text{sink}}(z) = (UT^{+\delta}(z), UT^{-\delta}(z)) = -T_{\text{source}}(z)$.

The algorithm looks for a directed path $P$ from $T_{\text{source}}(z)$ to $T_{\text{sink}}(z)$ in $R(x)$. To preserve skew symmetry of $x$, whenever we Augment path $P$ we also augment $-P$, which by skewness is also a directed path from $T_{\text{source}}(z)$ to $T_{\text{sink}}(z)$ in $R(x)$: Suppose, e.g. (the other cases are similar), that $P$ starts at $t \in UT^{-\delta}(z)$ (therefore $\text{sgn}(t) = +$ in order for $t$ to belong to $T_{\text{source}}$), and ends at $u \in UT^{+\delta}(z)$ (therefore $\text{sgn}(u) = +$ in order for $u$ to belong to $T_{\text{sink}}$); then $-P$ is the path from $-u \in T_{\text{source}}$ to $-t \in T_{\text{sink}}$. This $P$ and $-P$ look like the solid paths in Figure 2. Note that (in this case) since $z_u \geq +\delta$ and $\text{sgn}(u) = +$, for any $S \in \mathcal{R}_{\text{cur}}$ with $u \in S^+$ we have that $u \in W^z(S)$; similarly $z_{-t} \leq -\delta$ and $\text{sgn}(-t) = -$ imply that for any $S \in \mathcal{R}_{\text{cur}}$ with $-t \in S^-$ we have that $-t \in W^z(S)$. Therefore these augmentations have the side effect of getting rid of relatively large (w.r.t. $\delta$) violations of complementary slackness.

When we find such a skew pair of paths, Augment does a standard augmentation of $x$ on both of them by $\delta/2$ (we must use $\delta/2$ in case some or all of $P$ is self-skew, such as the dashed path in Figure 2). This reduces $||z||_{\text{cur}}$ by $\delta$, preserves $\delta$-feasibility, and does not affect $y$, $w$, nor $\varphi$. After each Augment we call routine ReduceV, which uses linear algebra to reduce the number of vertices in representation (10) to at most $n + 1$ (see [36] for a detailed description of ReduceV).

Note that augmenting paths are allowed to include only $x$-arcs, and cannot include arcs of $C$. However, suppose that in our search we discover a forward arc $t \to u \in C$ that we would like to use, but that the corresponding $t \to u \in A_{\text{relax}}$ is not in $R(x)$ because $x_{tu} > 0$. Then we do a FlowSwap operation: increase $\varphi_{tu}$ and $\varphi_{-u,-t}$ by $x_{tu}$ and reset $x_{tu}$ and $x_{-u,-t}$ to zero. Similarly, suppose that we cannot augment on $t \to u$ due to $x_{tu} > 0$ but that $\varphi_{ut} \geq x_{tu}$. Then we apply FlowSwap in a different way: decrease $\varphi_{ut}$ and $\varphi_{-t,-u}$ by $x_{tu}$ and reset $x_{tu}$ and $x_{-u,-t}$ to zero. Both versions keep $\partial^+\varphi + \partial^+ x$ invariant and so do not change $z$ (nor $y$, though $w$ changes; indeed, this is the only way that $\varphi$ ever changes), and allow $t \to u$ and $-u \to -t$ to join $R(x)$. The algorithms apply FlowSwaps as necessary to allow arcs of $C$ to implicitly participate in augmenting paths.

If no augmenting path exists, then let $S$ be the set of nodes in $\mathcal{S}_{\text{act}}$ that are reachable from $T_{\text{source}}(z)$ via directed paths in $R(x)$ (see Figure 3). By skew-symmetry, $-S$ is the set of nodes which have directed paths to $T_{\text{sink}}(z)$. FlowSwap implies that no arc of $C$ exits $S$. Since $t \in S \Rightarrow -t \in -S$, if $t$, $-t \in S$ we would have an immediate augmenting path, and so $S$ is a signed set. Therefore $S \cap \mathcal{S}_{\text{out}} = \emptyset$ (however, note that $(-S) \cap \mathcal{S}_{\text{out}}$ could be non-empty), and so $S \in \mathcal{R}_{\text{cur}}$. FlowSwaps also keep $\varphi$-flow into $S$ small (an optimal $S$ has zero $\varphi$-flow into it).

### 3.1.2 Changing $y$ When Blocked

Recall that $S$ induces subsets $G^i(S)$ and $B^i(S)$ of good and bad elements of $\text{abs}(\mathcal{S}_{\text{cur}})$ for each $i \in \mathcal{I}$. The algorithm moves bad elements to the far right of their orders and then flips their

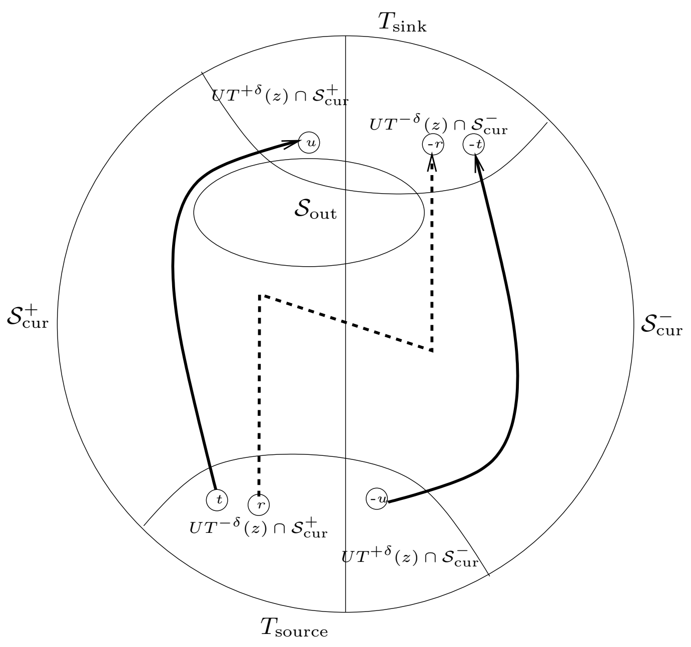
$T_{\text{sink}}$
$UT^{+\delta}(z) \cap \mathcal{S}^+_{\text{cur}}$
$UT^{-\delta}(z) \cap \mathcal{S}^-_{\text{cur}}$
$u$
-$r$
-$t$
$\mathcal{S}_{\text{out}}$
$\mathcal{S}^+_{\text{cur}}$
$\mathcal{S}^-_{\text{cur}}$
$t$
$r$
-$u$
$UT^{-\delta}(z) \cap \mathcal{S}^+_{\text{cur}}$
$UT^{+\delta}(z) \cap \mathcal{S}^-_{\text{cur}}$
$T_{\text{source}}$

Figure 2: Picture of the network used for augmentation with two possible skew pairs of augmenting paths (solid and dashed heavy paths; note that the dashed path is self-skew).

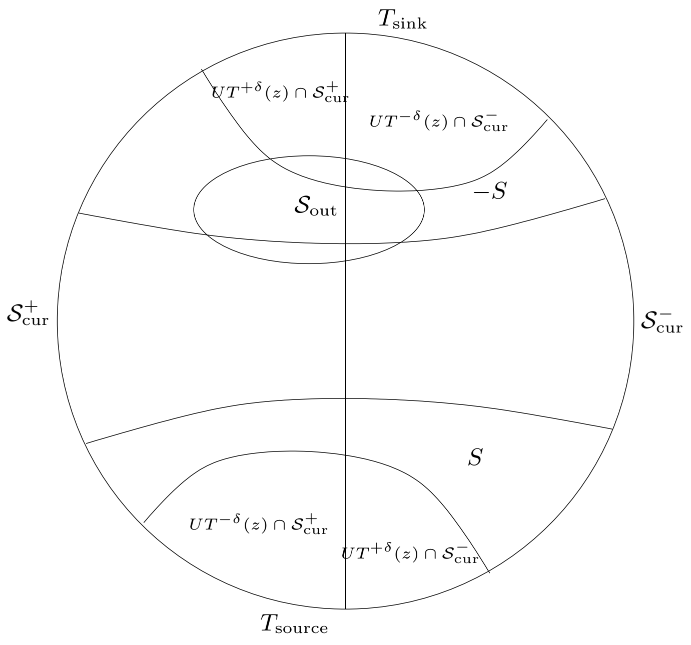
$T_{\text{sink}}$
$UT^{+\delta}(z) \cap \mathcal{S}^+_{\text{cur}}$
$UT^{-\delta}(z) \cap \mathcal{S}^-_{\text{cur}}$
$-S$
$\mathcal{S}_{\text{out}}$
$\mathcal{S}^+_{\text{cur}}$
$\mathcal{S}^-_{\text{cur}}$
$S$
$UT^{-\delta}(z) \cap \mathcal{S}^+_{\text{cur}}$
$UT^{+\delta}(z) \cap \mathcal{S}^-_{\text{cur}}$
$T_{\text{source}}$

Figure 3: Picture of the network used for augmentation with the reachable sets $S$ and $-S$.

signs (which makes them good), and moves good elements to the left of their orders. Thus there are two defects linear order $\prec^i$ might have that keep us from optimality: (1) a consecutive pair $u \prec^i t$ with either $t \in G^i(S)$ ($\Rightarrow t \in S$) and $u \notin G^i(S)$ ($t$ is a good element not at the left), or with $u \in B^i(S)$ ($\Rightarrow u \in -S$) and $t \notin B^i(S)$ ($u$ is a bad element not at the right), or (2) the right-most element $t$ of $\prec^i$ is in $B^i(S)$ (bad element $t$ is in position to get its sign flipped). In case (1) we call $(i, u, t)$ an *active triple*, and in case (2) we call $i, t$ a *tail-active pair*.

Given an active triple we apply procedure DOUBLE-EXCHANGE$(i, u, t)$. It generates a new pair $\prec^j$, $\sigma^j$, where $\prec^j$ matches $\prec^i$ except that $u$ and $t$ are swapped in $\prec^j$, so that $t \prec^j u$; it sets $\sigma^j = \sigma^i$. DOUBLE-EXCHANGE modifies (10) to include $v^j = v^{\prec^j, \sigma^j}$, and so we need to check whether it is consistent.

**Lemma 3.1** *Pair $\prec^j$, $\sigma^j$ in* DOUBLE-EXCHANGE *is consistent.*

**Proof:** Since $\prec^j$ contains the same signed elements as $\prec^i$, certainly they are also contained in $\mathcal{S}_{\text{act}}$. But if $t \to u \in C$, then $t \prec^j u$ would violate consistency. We have either $t \in S$, in which case $t \to u$ would be an arc of $C$ exiting $S$ which cannot happen, or $u \in -S$, in which case $t \to u$ is an arc of $C$ entering $-S$ which also cannot happen. ■

Given a tail-active pair we apply procedure TAIL-EXCHANGE$(i, t)$. It also generates a new pair $\prec^j$, $\sigma^j$, where $\sigma^j$ matches $\sigma^i$ except that $\sigma^j_t = -\sigma^i_t$; it sets $\prec^j = \prec^i$.

**Lemma 3.2** *Pair $\prec^j$, $\sigma^j$ in* TAIL-EXCHANGE *is consistent.*

**Proof:** Note that $t$ bad means that $-t \in S$. Flipping the sign of $t$ would violate consistency if $-t \notin \mathcal{S}_{\text{act}}$. This happens only if $-t \in \mathcal{S}_{\text{out}}$ meaning that $-t \to t \in C$, which would give an arc of $C$ exiting $S$ which cannot happen. If there were some $u \in \prec^j$, $u \neq t$, and $u \to -t \in C$ this would also violate consistency. But this implies that $u \in \prec^i$, and consistency of $\prec^i$ and $u \to -t \in C$ should imply that $-t \in \prec^i$, a contradiction. ■

**BSFM Subroutine** DOUBLE-EXCHANGE$(i, u, t)$
Applies when $u \prec^i t$ are consecutive in $\prec^i$, and ($t \in G^i(S)$ and $u \notin G^i(S)$, or $u \in B^i(S)$ and $t \notin B^i(S)$).

```
Set $\beta = v^i_u - [\hat{f}(u^{\prec^i} + u + t) - \hat{f}(u^{\prec^i} + t)]$.
Set $\alpha = \min(x_{tu}, \lambda_i \beta)$, and let $j$ be a new index.
If $\alpha = \lambda_i \beta \le x_{tu}$,    [ a *full* step ]
    Rename $v^i$ to $v^j$, add $\beta \text{sgn}(t)$ to $v^j_{\text{abs}(t)}$, subtract $\beta \text{sgn}(u)$ from $v^j_{\text{abs}(u)}$.
    Rename $\prec^i$, $\sigma^i$ to $\prec^j$, $\sigma^j$, swap $u$ and $t$ in $\prec^j$.
    Set $\lambda_j = \lambda_i$, add $j$ to $\mathcal{I}$, drop $i$ from $\mathcal{I}$.
Else    [ $\alpha = x_{tu} < \lambda_i \beta$, a *partial* step ]
    Set $v^j = v^i + \beta(\text{sgn}(t)\chi^{\text{abs}(t)} - \text{sgn}(u)\chi^{\text{abs}(u)})$.
    Set $\prec^j$ equal to $\prec^i$ with $u$ and $t$ swapped, $\sigma^j = \sigma^i$.
    Set $\lambda_i = \lambda_i - \alpha/\beta$, $\lambda_j = \alpha/\beta$, add $j$ to $\mathcal{I}$.
Set $x_{ut}$, $x_{-t,-u} = x_{ut} + \alpha$;
Reduce $x_{ut}$, $x_{tu}$, $x_{-t,-u}$, $x_{-u,-t}$ as necessary s.t. $x_{ut} \cdot x_{tu} = x_{-t,-u} \cdot x_{-u,-t} = 0$.
```

**BSFM Subroutine** TAIL-EXCHANGE$(i, t)$
Applies when $t$ is right-most in $\prec^i$, and $t \in B^i(S)$.

```
    Set $\beta = v^i_t + \hat{f}(t^{\prec^i} + (-t)) - \hat{f}(t^{\prec^i})$.
    Set $\alpha = \min(x_{-t,t}, \lambda_i\beta)$, and let $j$ be a new index.
    If $\alpha = \lambda_i\beta \leq x_{-t,t}$,    [ a *full* step ]
        Rename $v^i$ to $v^j$, subtract $\beta\text{sgn}(t)$ from $v^j_{\text{abs}(t)}$.
        Rename $\prec^i$, $\sigma^i$ to $\prec^j$, $\sigma^j$, flip sign of $\sigma^j_{\text{abs}(t)}$.
        Set $\lambda_j = \lambda_i$, add $j$ to $\mathcal{I}$, drop $i$ from $\mathcal{I}$.
    Else    [ $\alpha = x_{-t,t} < \lambda_i\beta$, a *partial* step ]
        Set $v^j = v^i - \text{sgn}(t)\beta\chi^{\text{abs}(t)}$.
        Set $\prec^j$ equal to $\prec^i$, and $\sigma^j = \sigma^i$ except with $\sigma^j_t = -\sigma^i_t$.
        Set $\lambda_i = \lambda_i - \alpha/\beta$, $\lambda_j = \alpha/\beta$, add $j$ to $\mathcal{I}$.
    Set $x_{t,-t} = x_{t,-t} + \alpha$, and reduce $x_{t,-t}$, $x_{-t,t}$ as necessary s.t. $x_{t,-t} \cdot x_{-t,t} = 0$.
```

To compute the new vertices $v^j$, from Signed Greedy and algebra note that for DOUBLE-EXCHANGE $v^j = v^i + \beta(\text{sgn}(t)\chi^{\text{abs}(t)} - \text{sgn}(u)\chi^{\text{abs}(u)})$, where $\beta = [\hat{f}(u^{\prec^i} + u) - \hat{f}(u^{\prec^i})] - [\hat{f}(u^{\prec^i} + u + t) - \hat{f}(u^{\prec^i} + t)] = v^i_u - [\hat{f}(u^{\prec^i} + u + t) - \hat{f}(u^{\prec^i} + t)]$. Note that $\beta \geq 0$ by (4). For TAIL-EXCHANGE $v^j = v^i - \text{sgn}(t)\beta\chi^{\text{abs}(t)}$, where $\beta = \hat{f}(t^{\prec^i} + t) + \hat{f}(t^{\prec^i} + (-t)) - 2\hat{f}(t^{\prec^i}) = v^i_t + \hat{f}(t^{\prec^i} + (-t)) - \hat{f}(t^{\prec^i})$. Note that $\beta \geq 0$ by (5). These $\beta$'s are sometimes called *exchange capacities*.

DOUBLE-EXCHANGE and TAIL-EXCHANGE then both try to replace $v^i$ by $v^j$ in (10). In order to keep $z$ invariant we need to change $x$ so that $y + \partial^+ x$ stays constant. Due to the upper bounds of $\delta$ on $x$ we may not be able to completely replace $v^i$ with $v^j$ in (10), so let us assume we replace $\lambda_i v^i$ by $(\lambda_i - \alpha/\beta)v^i + (\alpha/\beta)v^j$ in (10).

For DOUBLE-EXCHANGE, this would change $y$ by $\alpha(\text{sgn}(t)\chi^{\text{abs}(t)} - \text{sgn}(u)\chi^{\text{abs}(u)})$. We can counterbalance this by increasing $x_{ut}$ and $x_{-t,-u}$ by $\alpha$. Note that if $t \in G^i(S)$ then $t \in S$ and $u \notin S$. We must have $x_{tu} > 0$, else $t \to u$ would be in $R(x)$, implying that $u \in S$, a contradiction; similarly $t \in B^i(S)$ implies that $x_{tu} > 0$. Since $x_{ut} \cdot x_{tu} = 0$ we get $x_{ut} = 0$. To keep $x_{ut} \cdot x_{tu} = 0$ true, the first $\min(\alpha, x_{tu})$ of "increase to $x_{ut}$" would actually go towards reducing $x_{tu}$. For the analysis of the algorithm it is important that when $\alpha < \lambda_i\beta$ we choose $\alpha$ large enough that $x_{tu}$ hits zero. Hence we choose $\alpha = \min(x_{tu}, \lambda_i\beta)$. We note for later use in BSFM-FC in Section 3.5 that there is flexibility here: we could feasibly choose any value for $\alpha$ satisfying $x_{tu} \leq \alpha \leq \delta + x_{tu}$ without changing the algorithm's running time.

For TAIL-EXCHANGE, replacing $\lambda_i v^i$ by $(\lambda_i - \alpha/\beta)v^i + (\alpha/\beta)v^j$ in (10) would change $y$ by $-\alpha\text{sgn}(t)\chi^{\text{abs}(t)}$. We can counterbalance this by increasing (self-skew) $x_{t,-t}$ by $\alpha$. Note that $t \in B^i(S)$ means that $t \in -S$ and $-t \in S$. We must have $x_{-t,t} > 0$, else $-t \to t$ would be in $R(x)$, implying that $t \in S$, a contradiction. Therefore $x_{t,-t} = 0$, and by the same argument we choose $\alpha = \min(x_{-t,t}, \lambda_i\beta)$. Again for Section 3.5 there is flexibility here: any $\alpha$ satisfying $x_{-t,t} \leq \alpha \leq \delta + x_{-t,t}$ suffices.

Note that in both cases, if $\alpha = \lambda_i\beta$, then the new coefficient of $v^i$ in (10) is zero, and so we drop $v^i$ from $\mathcal{I}$. We call this a *full step* (*saturating* in [24]); otherwise ($\alpha = \delta < \lambda_i\beta$), we call it a *partial step* (*nonsaturating* in [24]). Note that for both DOUBLE-EXCHANGE and TAIL-EXCHANGE, a full step takes $O(\text{EO})$ time, and a partial step takes $O(\text{EO} + n)$ time.

## 3.2 Extending BRefine to BRefine$\mathcal{R}$

We now have all the ingredients we need to implement BRefine$\mathcal{R}$.

```
BSFM Subroutine BRefine$\mathcal{R}(\delta; y, x)$
Input: $y \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$, $2\delta$-feasible $x$

    Initialize $x \leftarrow x/2$, $z \leftarrow y + \partial^+\varphi + \partial^+ x$.
    Compute $UT^{-\delta}(z)$, $UT^{+\delta}(z)$, $T_{\text{source}}(z)$, $T_{\text{sink}}(z)$.
    Use FlowSwap to look for augmenting paths.
    While $\exists$ augmenting paths, or active triples or tail-active pairs, do
        If $\exists$ augmenting path $P$, Augment$(P, -P)$, ReduceV.
        Else ($\nexists$ such a path) if there is an active triple $(i, u, t)$ do
            Call Double-Exchange$(i, u, t)$.
        Else ($\nexists$ such a path, no active triple) if there is a tail-active pair $(i, t)$ do
            Call Tail-Exchange$(i, t)$.
        Update all data, renew FlowSwap augmenting path search.
    Return $S$ as an approximate optimal solution.
```

Theorem 2.2 says that an optimality condition for $T$ solving BSFM is to have a $w \in P_{\mathcal{R}_{\text{cur}}}(\hat{f})$ with $||w||_{\text{cur}} = -\hat{f}(T)$. We formally define $w$ and $x$ to be $\delta$-optimal if there is some $T \in \mathcal{R}_{\text{cur}}$ such that $w$ and $T$ satisfy $||w||_{\text{cur}} \le (2n^2 + 2n)\delta - \hat{f}(T)$, a relaxed version of this. The following lemma shows the approximate solution $S$ at the end of BRefine$\mathcal{R}$ proves that the final $w$ is $\delta$-optimal. (Many of our proofs are similar to those in [24], but we include the details for completeness.)

**Lemma 3.3** *When* BRefine$\mathcal{R}$ *ends,* $S$ *is in* $\mathcal{R}_{\text{cur}}$ *and is tight for* $y$*, and we have* $||w||_{\text{cur}} < (2n^2 + 2n)\delta - \hat{f}(S)$ *and* $||z||_{\text{cur}} < 2n\delta - \hat{f}(S)$*.*

**Proof:** (The $||z||_{\text{cur}}$ statement is similar to [24, Lemma 4.1].) We already argued that $S \in \mathcal{R}_{\text{cur}}$. Since no arc of $C$ exits $S$, $v^i(S) = \hat{f}(S)$ for all $i \in \mathcal{I}$, and so $y(S) = \hat{f}(S)$.

Note that $S^+ \subseteq \mathcal{S}^P_{\text{act}}$ and $S^- \subseteq \mathcal{S}^N_{\text{act}}$. Since $T_{\text{source}} \sqsubseteq S$, for $e \in S^+$ we have $z_e < +\delta$, so that $z^-(S^+) > z(S^+) - |S^+|\delta$; for $e \in S^-$ we have $z_e > -\delta$, so that $z^+(S^-) < z(S^-) + |S^-|\delta$. Since $S \sqsubseteq \mathcal{S}_{\text{act}} - T_{\text{sink}}$, for $e \in \mathcal{S}^P_{\text{act}} - S^+$ we have $z_e > -\delta$, so that $z^-(\mathcal{S}^P_{\text{act}} - S^+) > -|\mathcal{S}^P_{\text{act}} - S^+|\delta$; for $e \in \mathcal{S}^N_{\text{act}} - S^-$ we have $z_e < +\delta$, so that $z^+(\mathcal{S}^N_{\text{act}} - S^-) < |\mathcal{S}^N_{\text{act}} - S^-|\delta$.

When $u \to t \notin C$ interpret $\varphi_{ut}$ as 0. Then FlowSwap and the definition of $S$ ensure that for all $t \in S$ and $u \notin S$, $x_{tu} - \varphi_{ut} > 0$, implying that $\partial^+\varphi(S) + \partial^+ x(S) > 0$. Therefore $y(S) = z(S) - \partial^+ x(S) - \partial^+\varphi(S) < z(S)$. Hence $||z||_{\text{cur}} = z^+(S^-) - z^-(S^+) + z^+(\mathcal{S}^N_{\text{act}} - S^-) - z^-(\mathcal{S}^P_{\text{act}} - S^+) < (z(S^-) + |S^-|\delta) - (z(S^+) - |S^+|\delta) + |\mathcal{S}^N_{\text{act}} - S^-|\delta + |\mathcal{S}^P_{\text{act}} - S^+|\delta < -y(S) + \delta(|\mathcal{S}^P_{\text{act}}| + |\mathcal{S}^N_{\text{act}}|) \le -\hat{f}(S) + 2n\delta$.

Since $0 \le x \le \delta$, $\partial^+ x_e \le 2n\delta$. Due to $w = z - \partial^+ x$, $||w||_{\text{cur}} \le ||z||_{\text{cur}} + 2n^2\delta$, and so $||w||_{\text{cur}} < (2n^2 + 2n)\delta - \hat{f}(S)$. ■

**Theorem 3.4** BRefine$\mathcal{R}$ *runs in* $O(n^5\text{EO})$ *time.*

**Proof:** (Similar to [24, Lemmas 4.1, 4.3, and 4.4].) We first claim that there are $O(n^2)$ augmentations. The $x \leftarrow x/2$ step changes $||z||_{\text{cur}}$ by at most $2n^2\delta$, and so for the $T$ proving

$\delta$-optimality of $z$ the initial $z$ satisfies $||z||_{\text{cur}} \leq (2n+2n^2)\delta - \hat{f}(T)$. Now $||z||_{\text{cur}} \geq ||w||_{\text{cur}} - 2n^2\delta$, and $||w||_{\text{cur}} \geq -\hat{f}(T)$, so that $||z||_{\text{cur}} \geq -\hat{f}(T) - 2n^2\delta$. Each augmentation decreases $||z||_{\text{cur}}$ by $\delta$, and so there are at most $2n + 4n^2 = O(n^2)$ augmentations.

Note that $|\mathcal{I}|$ increases by one only at a partial step, and each partial step causes at least one new node to get added to $S$, so there can be only $O(n)$ partial steps before an augmentation, showing that $|\mathcal{I}|$ is $O(n)$. Each $i \in \mathcal{I}$ can have only $O(n^2)$ Double-Exchanges and $O(n)$ Tail-Exchanges applied to it between augmentations, for a total of $O(n^3)$ Double-Exchanges and $O(n^2)$ Tail-Exchanges per augmentation. Each call to Double-Exchange and Tail-Exchange involves two evaluation oracle calls. ■

## 3.3 Extending FuI-BSFM to BSFM on Signed Ring Families

We can now easily plug BRefine$\mathcal{R}$ in place of BRefine in FuI-BSFM to get a weakly polynomial BSFM algorithm that works on signed ring families. The outer framework is to use Signed Greedy to compute a vertex $v$ and call BRefine$\mathcal{R}$ until $\delta$ is small enough that we can conclude exact optimality.

**Weakly Polynomial BSFM Algorithm for Signed Ring Families**

```
Initialize by choosing $\prec^1$, $\sigma^1$ to be any consistent pair, $y = v^1$, and $\mathcal{I} = \{1\}$.
Compute $\delta^0 = \max_{t \in \mathcal{S}_{\text{act}}}\{\hat{f}(D_t) - \hat{f}(D_t - t)\}$, initialize $\delta = \delta^0$ and $x = 0$.
While $\delta \geq 1/3n^2$,  [ when $\delta < 1/3n^2$ current $S$ is optimal ]
    Set $\delta \leftarrow \delta/2$.
    Call BRefine$\mathcal{R}$. [ converts $2\delta$-optimality to $\delta$-optimality ]
Return last approximate solution $S$ from BRefine$\mathcal{R}$ as an optimal BSFM solution.
```

We now prove that this works.

**Theorem 3.5** *When $\hat{f}$ is integer-valued, the BSFM Algorithm solves BSFM over the signed ring family $\mathcal{R}_{\text{cur}}$ in $O(n^5\text{EO}\log(nM))$ time.*

**Proof:** (Similar to [24, Theorems 4.2 and 4.5].) Lemma 3.3 shows that the $S$ produced at the end of one call to BRefine$\mathcal{R}$ proves that the initial $z$ at the next call to BRefine$\mathcal{R}$ is $2\delta$-optimal.

If a call to BRefine$\mathcal{R}$ ends with $S \in \mathcal{R}_{\text{cur}}$ and $\delta$-optimal $w$ with $\delta < 1/3n^2$, then Lemma 3.3 shows that $||w||_{\text{cur}} \leq (2n^2 + 2n)\delta - \hat{f}(S) < 3n^2\delta - \hat{f}(S) < 1 - \hat{f}(S)$. By (8), for any $T \in \mathcal{R}_{\text{cur}}$ we have $||w||_{\text{cur}} \geq -\hat{f}(T)$, and so $1 - \hat{f}(S) > -\hat{f}(T)$. Since $\hat{f}$ is integer-valued, $S$ solves BSFM.

By Lemma 2.3, for $y^0 = w^0$ as the initial values of $y$ and $w$, we have $||w^0||_{\text{cur}} \leq 2n\delta^0 = 2n\delta^0 - \hat{f}(\emptyset)$, and so $w^0$ is $\delta^0$-optimal. Clearly $\delta^0 \leq 2M$. Since each call to BRefine$\mathcal{R}$ halves $\delta$, there are at most $\log_2(6Mn^2) = O(\log(nM))$ such calls. Theorem 3.4 shows that each call costs $O(n^5\text{EO})$ time, for a total of $O(n^5\text{EO}\log(nM))$ time. ■

When the initial $C$ is the empty set (so that $\mathcal{R}_{\text{cur}} = 3^E$ and $||y^0||_{\text{cur}} = ||y^0||_1$) we can instead initialize $\delta^0 = ||y^0||_1/n^2$ and sharpen this slightly to match the bound on FuI-BSFM from [24].

**Corollary 3.6** *If $\hat{f}$ is integer-valued and $\mathcal{R}_{\text{cur}} = 3^E$, then this algorithm solves BSFM in $O(n^5\text{EO}\log M)$ time.*

**Proof:** Define $S^+ = \{e \mid y^0_e > 0\}$ and $S^- = \{e \mid y^0_e < 0\}$. Then $||y^0||_1 = y^0(S) \le \hat{f}(S) \le M$, and so $y^0$ is $\delta^0$-optimal. Hence now we need at most $\log_2(3M) = O(\log M)$ calls to BRefine$\mathcal{R}$. ■

## 3.4 Extending IFF-SP to BSFM

The challenge in making a weakly polynomial scaling algorithm like the IFF Algorithm strongly polynomial is to avoid having to call BRefine$\mathcal{R}$ for each scaled value of $\delta$, since the weakly polynomial factor $O(\log M)$ is really $\Theta(\log M)$. The rough idea is to find a way for the current data of the problem to reveal a good starting value of $\delta$, and then to apply $O(\log n)$ calls to BRefine$\mathcal{R}$ to get close enough to optimum that we can "fix a variable", which can happen only a strongly polynomial number of times. Letting the current data determine the value of $\delta$ can also be seen as a way to allow the algorithm to make much larger decreases in $\delta$ than would be available in the usual scaling framework.

The general mechanism for fixing a variable is to prove a "proximity lemma" as in Tardos [50] that says that if the value of a variable gets too far from a bound, then we can remove that bound, and then reduce the size of the problem. The proximity lemma below says that if component $w_e$ is negative enough w.r.t. $\delta$, then $+e$ belongs to $\overline{S}$ for every minimizer $\overline{S}$ of $\hat{f}$, and if $w_e$ is positive enough w.r.t. $\delta$, then $-e$ belongs to $\overline{S}$ for every minimizer $\overline{S}$ of $\hat{f}$. A strongly polynomial number of such steps would bring us to optimality.

**Lemma 3.7** *After a call to* BRefine$\mathcal{R}$*, if there is some $e \in \mathcal{S}^P_{\text{act}}$ such that the current $w$ satisfies $w_e < -3n^2\delta$, then $+e$ belongs to $\overline{S}$ for every minimizer $\overline{S}$ of $\hat{f}$; if there is some $e \in \mathcal{S}^N_{\text{act}}$ such that the current $w$ satisfies $w_e > +3n^2\delta$, then $-e$ belongs to $\overline{S}$ for every minimizer $\overline{S}$ of $\hat{f}$.*

**Proof:** Let $\overline{S}$ be any solution to BSFM for $\hat{f}$. Recall that $||w||_{\text{cur}} = w^+(\mathcal{S}^N_{\text{act}}) - w^-(\mathcal{S}^P_{\text{act}})$. By Lemma 3.3, at the end of a $\delta$-scaling phase, for the current approximate solution $S$ (note that $S$, $\overline{S} \in \mathcal{R}_{\text{cur}}$), we have $-||w||_{\text{cur}} = w^-(\mathcal{S}^P_{\text{act}}) - w^+(\mathcal{S}^N_{\text{act}}) \ge \hat{f}(S) - (2n^2 + 2n)\delta \ge \hat{f}(S) - 3n^2\delta$. Since $\overline{S}$ solves BSFM, we have $\hat{f}(S) \ge \hat{f}(\overline{S}) \ge w(\overline{S}^+) - w(\overline{S}^-) \ge w^-(\overline{S}^+) - w^+(\overline{S}^-)$. This implies that $w^-(\mathcal{S}^P_{\text{act}} - \overline{S}^+) - w^+(\mathcal{S}^N_{\text{act}} - \overline{S}^-) \ge -3n^2\delta$. Then if $e \in \mathcal{S}^P_{\text{act}}$ and $w_e < -3n^2\delta$ but $+e \notin \overline{S}^+$, we could add $-w_e > 3n^2\delta$ to this to get $w^-((\mathcal{S}^P_{\text{act}} - \overline{S}^+) - e) - w^+(\mathcal{S}^N_{\text{act}} - \overline{S}^-) > 0$, a contradiction, so we must have $e \in \overline{S}^+$. A similar argument establishes the case where $e \in \mathcal{S}^N_{\text{act}}$ and $w_e > +3n^2\delta$. ■

In order to use Lemma 3.7 to generate condition arcs we apply BRefine$\mathcal{R}$ to a bisubmodular function $\hat{f}_t$ (defined below) that is defined so that solving BSFM for $\hat{f}_t$ solves BSFM$_t$ for $\hat{f}$. If a set contains $t$, it must also contain all signed elements in $D_t$. Furthermore, it cannot contain $-t$, and so it also cannot contain any of the signed elements in $A_{-t} = -D_t$. The set of active (current) nodes for $\hat{f}_t$ is $\mathcal{S}_{\text{act}}(t)$ ($\mathcal{S}_{\text{cur}}(t)$), which is the nodes in $\mathcal{S}_{\text{act}}$ ($\mathcal{S}_{\text{cur}}$) with the nodes in $D_t \cup A_{-t}$ deleted ($t \notin \mathcal{S}_{\text{out}}$ implies that $D_t$ is disjoint from $A_{-t}$, so we can use ordinary union here). Then the signed ring family $\mathcal{R}_{\text{cur}}(t)$ is the family of reduced closed sets of $(\mathcal{S}_{\text{cur}}(t), C)$ (where arcs incident to $\mathcal{S}_{\text{cur}} - \mathcal{S}_{\text{cur}}(t)$ are also deleted). For $T \in \mathcal{R}_{\text{cur}}(t)$ define $\hat{f}_t(T) = \hat{f}(T \sqcup D_t) - \hat{f}(D_t)$ (here $T$ is disjoint from $D_t$, so this is just a signed version of an ordinary union).

Define $D_u(t)$ to be the descendents of $u$ in $(\mathcal{S}_{\text{cur}}(t), C)$. Recall that $\delta^0 = \max_{u \in \mathcal{S}_{\text{act}}}\{\hat{f}(D_u) - \hat{f}(D_u - u)\}$. Then Lemma 2.3 would be true for $\hat{f}_t$ if we replaced the $\delta^0$ for $\hat{f}$ with $\delta^0(t) = \max_{u \in \mathcal{S}_{\text{cur}}(t)}\{\hat{f}_t(D_u(t)) - \hat{f}_t(D_u(t) - u)\} = \max_{u \in \mathcal{S}_{\text{cur}}(t)}\{\hat{f}(D_u(t) \sqcup D_t) - \hat{f}((D_u(t) - u) \sqcup D_t)\}$.

Since $D_u \sqsubseteq (D_u(t) \sqcup D_t)$, we get by (4) that $\delta^0(t) \leq \delta^0$, and so the conclusion of Lemma 2.3 remains true for $\hat{f}_t$ and the original $\delta^0$. Similarly, Lemmas 3.3 and 3.7 apply to $\hat{f}_t$ using the original $n$.

Function $\hat{f}_t$ is clearly again bisubmodular on $\mathcal{R}_{\text{cur}}(t)$ with $\hat{f}(\emptyset) = 0$, and solving BSFM on $\hat{f}_t$ would yield a signed set with smallest function value among all sets containing $t$. Therefore if Lemma 3.7 tells us that $u$ belongs to every minimizer of $\hat{f}_t$, then we could add the condition arc $t \to u$ to $C$ (and so also $-u \to -t$, by Lemma 2.1).

We choose a $t$ such that $\delta^0 = \hat{f}(D_t) - \hat{f}(D_t - t)$, and then subroutine Fix calls BRefine$\mathcal{R}$ on $\hat{f}_t$ $O(\log n)$ times to produce some approximate solution with $\delta < \delta^0/3n^3$. We call a signed set $T \in \mathcal{R}_{\text{cur}}(t)$ with $\hat{f}_t(T) \leq -\delta^0$ *highly negative* (for $\hat{f}_t$). If there is a highly negative set when we call Fix, then for the $w \in P_{\mathcal{R}_{\text{cur}}(t)}(\hat{f}_t)$ at the end of Fix we have $w(T) \leq \hat{f}(T) \leq -\delta^0 < -3n^3\delta$. This implies that there is at least one $e \in T^+$ with $w_e \leq -3n^2\delta$ or $e \in T^-$ with $w_e \geq +3n^2\delta$. We call such an $e$ a *highly far element*. If $w_e \leq -3n^2\delta$ then we call $e$ *highly negative* and Lemma 3.7 then shows that $+e$ belongs to every minimizer of $\hat{f}_t$ and so we can add condition arc $t \to +e$ (and $-e \to -t$) to $C$; similarly if $w_e \geq +3n^2\delta$ then we call $e$ *highly positive* and we can add condition arc $t \to -e$ (and $+e \to -t$) to $C$. Since $+e, -e \notin D_t$, this represents real progress. Fix checks for highly far elements after the calls to BRefine$\mathcal{R}$ and adds such arcs to $C$.

Unfortunately it is difficult to guarantee that a highly negative set exists. IFF [33] go to some trouble to manufacture such a set, and rely on the fact that $y(E) = g(E)$ for all $y$ in the base polytope of the submodular function $g$. In the bisubmodular case, in general $w \in P(\hat{f})$ does not satisfy such an equation, and so it is not clear how to accomplish this. This is the hurdle that prevented [24] from achieving a strongly polynomial BSFM algorithm. Instead we take a "lazy" approach developed in [36] that does not depend on the existence of such a set, but rather concludes that $t$ cannot belong to an optimal solution if no element satisfying the condition of Lemma 3.7 appears.

**Lemma 3.8** *If a call to* Fix *produces no element satisfying the conditions of Lemma 3.7 with $\hat{f}$ replaced by $\hat{f}_t$, then no optimal solution contains $t$.*

**Proof:** If a highly negative set existed when Fix was called, then such an element would certainly be produced. Thus no highly negative set exists, and so we must have that every $T \in \mathcal{R}_{\text{cur}}(t)$ has $\hat{f}_t(T) > -\delta^0$. Expanding both sides gives $\hat{f}(T \cup D_t) - \hat{f}(D_t) > -(\hat{f}(D_t) - \hat{f}(D_t - t))$, or $\hat{f}(T \cup D_t) > \hat{f}(D_t - t)$. Then $T \cup D_t$ is a generic feasible set containing $t$, and $D_t - t$ is a specific set not containing $t$. This proves that $t$ cannot belong to any solution to BSFM. ■

Lemma 3.8 shows that when Lemma 3.7 does not apply after Fix, we can add $t$ to $\mathcal{S}_{\text{out}}$ (by adding $t \to -t$ to $C$), which is again real progress. As the algorithm proceeds we are getting rid of "big" elements, and so $\delta^0$ tends towards zero (in fact it is not hard to show that $\delta^0$ is non-increasing). We need to know how to construct an optimal BSFM solution if we attain $\delta^0 \leq 0$, and the next lemma shows how to do this.

**Lemma 3.9** *If $\delta^0 \leq 0$, then for any consistent pair $\prec^0$, $\sigma^0$, $\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^0$ solves BSFM for $\hat{f}$, with dual optimal solution $y = v^{\prec^0,\sigma^0}$.*

**Proof:** Define $S = (S^+, S^-) = (\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^0)$. From Lemma 2.3 we have that for each $g \in S^+$, $y_g \leq \delta^0 \leq 0$, and for each $h \in S^-$, $y_h \geq -\delta^0 \geq 0$. Therefore $y^-(\mathcal{S}^P_{\text{act}}) = y^-(S^+) = y(S^+)$ and $y^+(\mathcal{S}^N_{\text{act}}) = y^+(S^-) = y(S^-)$.

Now (7) implies that $y(S) = \hat{f}(S)$. Thus we get that $-||y||_{\text{cur}} = y^+(\mathcal{S}^N_{\text{act}}) - y^-(\mathcal{S}^P_{\text{act}}) = y(S) = \hat{f}(S)$. This implies that $y$ and $S$ are jointly optimal. ∎

**Subroutine** $\textsc{Fix}(\hat{f}_t, (\mathcal{S}_{\text{cur}}(t), C), \delta^0)$
Applies to $\hat{f}_t$ defined on reduced closed sets of $(\mathcal{S}_{\text{cur}}(t), C)$, and $v_u \le \delta^0$ for all vertices $v$ of $P_{\mathcal{R}_{\text{cur}}(t)}(\hat{f}_t)$.
Returns $\mathcal{N} \subseteq \mathcal{S}_{\text{act}}(t)$ of elements in every optimal solution and their descendents.

```
    Initialize $\prec, \sigma$ as any pair consistent with $C$, $y \leftarrow v^{\prec,\sigma}$, $\delta \leftarrow \delta^0$, and $\mathcal{N} = \emptyset$.
    While $\delta \ge \delta^0/n^3$ do
        Set $\delta \leftarrow \delta/2$.
        Call BRefine$\mathcal{R}$.
    For $e \in \text{abs}(\mathcal{S}_{\text{act}}(t))$ do [ Lemma 3.7: add descendents of highly far nodes to $\mathcal{N}$ ]
        If $w_e < -3n^2\delta$ set $\mathcal{N} \leftarrow \mathcal{N} \cup D_{+e}$.
        If $w_e > +3n^2\delta$ set $\mathcal{N} \leftarrow \mathcal{N} \cup D_{-e}$.
    Return $\mathcal{N}$.
```

When $\delta^0 > 0$, the proof of Theorem 3.5 showed that the initial $w^0$ is $\delta^0$-optimal. When we then apply $\textsc{Fix}$ to $\hat{f}_t$, if it finds highly far elements then we add new arcs to $C$; if it finds no highly far elements, then we add $t$ to $\mathcal{S}_{\text{out}}$.

**Strongly Polynomial BSFM Algorithm (BSFM-SP)**

```
    Initialize $\mathcal{S}_{\text{out}} \leftarrow \emptyset$, $C \leftarrow \emptyset$, $\mathcal{S}_{\text{cur}} \leftarrow \overline{E}$.
    While abs($\mathcal{S}_{\text{cur}}$) > 1 do
        Compute $\delta^0 = \max_{u \in \mathcal{S}_{\text{act}}}\{\hat{f}(D_u) - \hat{f}(D_u - u)\}$ and $t \in \mathcal{S}_{\text{act}}$ as an argmax.
        If $\delta^0 \le 0$ then [ implement Lemma 3.9 ]
            For consistent $\sigma^0$ return $\overline{E}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^0)$ as a BSFM solution.
        Else [ $\delta^0 > 0$ ] do
            Set $\mathcal{N} \leftarrow \textsc{Fix}(\hat{f}_t, (\mathcal{S}_{\text{cur}}(t), C), \delta^0)$.
            If $\mathcal{N} = \emptyset$, add $t \to -t$ to $C$. [ add $t$ to $\mathcal{S}_{\text{out}}$ ]
            Else [ $\mathcal{N} \ne \emptyset$ ] do
                For all $u \in \mathcal{N}$ add $t \to u$ and $-u \to -t$ to $C$. [ Lemma 2.1 ]
            Call UpdateS, update all $D_s$'s, $A_s$'s.
    Return whichever of $\emptyset$ and $\overline{E}(\mathcal{S}_{\text{act}})$ has a smaller function value.
```

We now prove that this is correct, picking out the main points of the proof in boldface.

**Theorem 3.10** *Algorithm BSFM-SP is correct, and runs in $O(n^7 \text{EO} \log n)$ time.*

**Proof:**

**When a highly negative $\mathcal{T} \in \mathcal{D}_t$ exists, a call to $\textsc{Fix}(\hat{f}_t, (\mathcal{S}_{\text{cur}}(t), C), \delta^0)$ results in at least one element added to $\mathcal{N}$:** The call to $\textsc{Fix}$ reduces $\delta$ from $\delta^0$ to $\delta^0/3n^3$. Then $\mathcal{T}$ highly negative and $\mathcal{T} \in \mathcal{D}_t$ imply that $w(\mathcal{T}) \le y(\mathcal{T}) \le \hat{f}(\mathcal{T}) \le -\delta^0 = -3n^3\delta$. This implies that there is at least one highly far $s \in \mathcal{S}_{\text{act}}$, so at least one element gets added to $\mathcal{N}$.

**If** $\text{Fix}(\hat{f}_t, (\mathcal{S}_{\text{cur}}(t), C), \delta^0)$ **finds no highly negative element, then we can add** $t$ **to** $\mathcal{S}_{\text{out}}$**:** By Lemma 3.8.

**The algorithm returns a solution to BSFM:** If some $\delta^0 \le 0$, then we showed above that the returned set is optimal. Otherwise the algorithm terminates because $|\text{abs}(\mathcal{S}_{\text{cur}})| \le 1$. In this case the only two choices left for solving BSFM are $\mathcal{S}_{\text{act}}$ and $\emptyset$, and the algorithm returns the better of these.

Fix **calls** BRefine$\mathcal{R}$ $O(\log n)$ **times:** Parameter $\delta$ starts at $\delta^0$, ends at $\delta^0/n^3$, and is halved at each iteration. Thus there are $\log_2(2n^3) = O(\log n)$ calls to BRefine$\mathcal{R}$.

**The algorithm calls** Fix $O(n^2)$ **times:** Each call to Fix adds at least one arc to $C$ (either via $\mathcal{S}_{\text{out}}$ or $\mathcal{N}$). Since there are only $2n(2n-1)$ possible arcs for $C$, Fix is called $O(n^2)$ times.

**The algorithm runs in** $O(n^7\text{EO}\log n)$ **time:** Theorem 3.4 says that one call to BRefine$\mathcal{R}$ costs $O(n^5\text{EO})$ time. Each call to Fix calls BRefine$\mathcal{R}$ $O(\log n)$ times, and the algorithm calls Fix $O(n^2)$ times, for a total time of $O(n^7\text{EO}\log n)$. ∎

## 3.5 Extending I-FC to BSFM

Recall that a fully combinatorial algorithm cannot use multiplication or division, and must also be strongly polynomial. This implies that it cannot call ReduceV, since the linear algebra in ReduceV apparently needs to use multiplication and division in a way that cannot be simulated with addition and subtraction. This suggests that we adapt BSFM-SP by avoiding the calls to ReduceV; this degrades the running time since $|\mathcal{I}|$ is allowed to get much larger than $n$, but the algorithm will work as long as we can show that $|\mathcal{I}|$ remains polynomially-bounded.

BSFM-SP adds new $v^j$'s only at partial steps, and only one new $v^j$ at a time. Since there are at most $n$ partial steps per Augment, this means that each Augment creates at most $n$ new $v^j$'s. In BSFM-SP, each call to Fix calls BRefine$\mathcal{R}$ $O(\log n)$ times. Each call to BRefine$\mathcal{R}$ does $O(n^2)$ Augments, or a total of $O(n^2 \log n)$ Augments for each call to Fix, for a total of $O(n^3 \log n)$ $v^j$'s added in each call to Fix. Each call to Fix starts out with $|\mathcal{I}| = 1$, so $|\mathcal{I}|$ stays bounded by $O(n^3 \log n)$ when we do not use ReduceV.

Without ReduceV, each of the $O(n^3 \log n)$ vertices in $\mathcal{I}$ has $O(n^2)$ possible active triples (each requiring $O(\text{EO})$ work), so now the work from full steps between each Augment is $O(n^5 \log n \cdot \text{EO})$. Multiplied by the $O(n^2)$ Augments, this gives $O(n^7 \log n \cdot \text{EO})$ as the time for BRefine$\mathcal{R}$. Multiplied by the $O(\log n)$ calls to BRefine$\mathcal{R}$ per call to Fix, and by the $O(n^2)$ calls to Fix overall, we would get a total of $O(n^9 \log^2 n \cdot \text{EO})$ time for the algorithm without calling ReduceV.

However, getting rid of ReduceV is not sufficient to make BSFM-SP fully combinatorial, as we must deal with its various other multiplications and divisions. The only non-trivial remaining multiplications and divisions are the terms $\lambda_i\beta$ and $\alpha/\beta$ that arise in Double-Exchange and Tail-Exchange, and the $\delta^0/n^3$ and $n^2\delta$ that arise in Fix.

### 3.5.1 Scaling via Doubling

Below we modify the representation (10) by implicitly multiplying through by a common denominator so that each $\lambda_i$ is an integer bounded by a polynomial in $n$. Then $\lambda_i\beta$ can be dealt with using repeated addition. The common denominator is implemented by changing from halving $\delta$ at each iteration to doubling a scaling parameter. We need another factor of $n$ for technical reasons, so we instead compute $n^4$. This can be done via $O(n)$ repeated additions. Expressions like $n^2\delta$ can also be dealt with using repeated addition in $O(n)$ time.

We plan to simulate a discrete version of $\alpha/\beta$ via repeated subtractions. To do this we need to know that the quotient has strongly polynomial size in terms of the scale factor. Recall that $\alpha$ was defined w.r.t. flow on a *key arc*, which is $t \to u$ for Double-Exchange, and is $-t \to t$ for Tail-Exchange; denote this key arc by $r \to s$. Then the prior definition was $\alpha = \min(x_{rs}, \lambda_i\beta)$. As previously noted, we could choose any $\alpha$ satisfying $x_{rs} \le \alpha \le x_{rs} + \delta$ without changing the analysis of the algorithm. This gives us enough flexibility to discretize the quotient. Indeed, this is essentially what Iwata [30] does.

BSFM-FC adapts BSFM-SP as follows: We denote corresponding variables in BSFM-FC by tildes, so where BSFM-SP has $x$, $y$, $z$, $\lambda$, $\delta$, etc., BSFM-FC has $\tilde{x}$, $\tilde{y}$, $\tilde{z}$, $\tilde{\lambda}$, $\tilde{\delta}$, etc. Recall from (10) that BSFM-SP keeps $y \in P_{\mathcal{R}_{\text{cur}(t)}}(\hat{f}_t)$ as a convex combination of vertices $y = \sum_{i\in\mathcal{I}} \lambda_i v^i$. The $\lambda_i$ satisfy $\lambda_i \ge 0$ and $\sum_{i\in\mathcal{I}} \lambda_i = 1$, but are otherwise arbitrary. To make the arithmetic discrete in BSFM-FC, we keep a scale factor $\text{SF} = 2^a$ (for $a$ a non-negative integer). We now keep each $\lambda_i$ as a fraction with integer numerator, and denominator SF. To clear the fractions we represent $\tilde{y}$ as $\text{SF}y \in P(\text{SF}\hat{f})$ and $\tilde{\lambda}_i = \text{SF}\lambda_i$, so that $\tilde{y} = \sum_{i\in\mathcal{I}} \tilde{\lambda}_i v^i$ with each $\tilde{\lambda}_i$ a positive integer, and $\sum_{i\in\mathcal{I}} \tilde{\lambda}_i = \text{SF}$. At the beginning of each call to Fix, as before we choose an arbitrary $\prec^1$ consistent with $\mathcal{D}$ and set $\tilde{y} = v^1$. Thus we choose $a = 0$, $\text{SF} = 2^0 = 1$, and $\tilde{\lambda}_1 = 1$ to satisfy this initially.

BSFM-SP starts each call to Fix with $\delta = \delta^0$ and halves it before each call to BRefine$\mathcal{R}$. BSFM-FC starts with $\tilde{\delta} = 2n\delta^0$, and instead of halving it, BSFM-FC doubles SF (increases $a$ by 1). This extra factor of $2n$ is needed to make Lemma 3.11 work, which in turn is needed to make the fully combinatorial discrete approximation of $\alpha/\beta$ lead to a $\tilde{\delta}$-feasible update to $\tilde{x}$. Even if we started out with $f$ defined on $3^E$, the proof of Lemma 3.11 forces us to ensure that only vertices consistent with $\mathcal{S}_{\text{act}}(t)$ are generated (the "explicit method" of handling $\mathcal{S}_{\text{act}}(t)$; see [36] for details).

### 3.5.2 Details of BSFM-FC

Lemma 3.11 below also needs that for each $v^i$ in the current representation of $y$, $\hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i) > -\delta^0$, which necessitates changing BSFM-SP: At the point that Double-Exchange or Tail-Exchange generates a new $v^j$ we check whether $\hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^j) > -\delta^0$. If it is not, it is highly negative, and we call Fix directly on $\hat{f}$ (instead of $\hat{f}_t$) to find some $s \in \mathcal{S}_{\text{act}}$ that is contained in all BSFM solutions via Lemma 3.7, and then we add $\overline{E}(D_s)$ to a set In of elements *in* all BSFM solutions. We then delete $D_s$ from $\mathcal{S}_{\text{act}}$ and re-set $\hat{f} \leftarrow \hat{f}_s$. Since each such highly negative call to Fix results in at least one new element of In, we can have only $O(n)$ of these overall, and this change does not impair the running time of the algorithm. This also means that we need the same sort of bound for $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$.

**Lemma 3.11** *If every $v^i$ in the current representation of $y$ has $\hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i) > -\delta^0$, then for any two vertices $v^i$ and $v^j$ of $P_{\mathcal{R}_{\text{cur}(t)}}(\hat{f}_t)$ and any $s \in \mathcal{S}_{\text{act}}(t)$, $|v^i_s - v^j_s| \le \tilde{\delta}$. In particular $\beta \le \tilde{\delta}$ in $P_{\mathcal{R}_{\text{cur}(t)}}(\hat{f}_t)$ (and also $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$).*

**Proof:** Note that for both Double-Exchange and Tail-Exchange $\beta$ equals $|v^i_e - v^j_e|$ for some $e$, so it suffices to prove the first statement, and we can assume that $y = v^i$ for some vertex $v^i$. We have that $y(\text{abs}(\mathcal{S}_{\text{act}}(t))|\sigma^i) = \hat{f}_t(\text{abs}(\mathcal{S}_{\text{act}}(t))|\sigma^i) = \hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i) - \hat{f}(D_t)$. Then

$\hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i) > -\delta^0$ and $\hat{f}(D_t) \le \sum_{s \in D_t} |\hat{f}(D_s) - \hat{f}(D_s - s)| \le |D_t|\delta^0$ imply that

$$y(\text{abs}(\mathcal{S}_{\text{cur}}(t))|\sigma^i) \ge -(|D_t| + 1)\delta^0. \tag{12}$$

For all $g \in \text{abs}(\mathcal{S}_{\text{act}}(t))$, either $g \in \mathcal{S}^P_{\text{act}}$ if $\sigma^i_g = +$, or $g \in \mathcal{S}^N_{\text{act}}$ if $\sigma^i_g = -$. Suppose that $\sigma^i_e = +$. Then Lemma 2.3 shows that $y_e \le \delta^0$. For all $g \in \text{abs}(\mathcal{S}_{\text{act}}(t))$ with $g \ne e$ and $\sigma^i_g = +$ add $-y_g \ge -\delta^0$ to (12), and for all $g \in \text{abs}(\mathcal{S}_{\text{act}}(t))$ with $g \ne e$ and $\sigma^i_g = -$ add $y_g \ge -\delta^0$ to (12), to get $y_e \ge -n\delta^0$. Thus $-n\delta^0 \le y_e \le \delta^0$ for any $e \in \text{abs}(\mathcal{S}_{\text{act}}(t))$ with $\sigma^i_e = +$, and similarly $-\delta^0 \le y_e \le n\delta^0$ for any $e \in \text{abs}(\mathcal{S}_{\text{act}}(t))$ with $\sigma^i_e = -$. Thus $\beta$ is at most $2n\delta^0 = \tilde{\delta}$. A simpler version of the same proof works for $P_{\mathcal{R}_{\text{cur}}}(\hat{f})$. ■

**BSFM Fully Combinatorial Algorithm (BSFM-FC)**

```
Initialize In ← $\emptyset$, $\mathcal{S}_{\text{out}}$, $C$, and $\mathcal{S}_{\text{act}}$.
While $|\mathcal{S}_{\text{act}}| > 1$ do
    Compute $\delta^0 = \max_{s \in \mathcal{S}_{\text{act}}}\{\hat{f}(D_s) - \hat{f}(D_s - s)\}$ and let $t \in \mathcal{S}_{\text{act}}$ be an argmax.
    If $\delta^0 \le 0$ then [ implement Lemma 3.9 ]
        For consistent $\sigma^0$ return $\overline{E}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^0)$ as a BSFM solution.
    Else [ $\delta^0 > 0$ ] do
        If some $v^i$ has $\hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i) \le -\delta^0$ do [ highly negative ]
            Set $\mathcal{N} \leftarrow$ Fix$(\hat{f}, (\mathcal{S}_{\text{cur}}, C), \delta^0)$.
            For each $s \in \mathcal{N}$ add $\overline{E}(D_s)$ to In, and re-set $\mathcal{S}_{\text{act}} \leftarrow \mathcal{S}_{\text{act}} - D_s$, $\hat{f} \leftarrow \hat{f}_s$.
        Else [$\delta^0 > 0$ and $\hat{f}(\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^i) > -\delta^0$ for all $i \in \mathcal{I}$] do
            Set $\mathcal{N} \leftarrow$ Fix$(\hat{f}_t, (\mathcal{S}_{\text{cur}}(t), C), \delta^0)$.
            If $\mathcal{N} = \emptyset$, set $\mathcal{S}_{\text{out}} \leftarrow \mathcal{S}_{\text{out}} \cup \overline{E}(A_t)$.
            Else [ $\mathcal{N} \ne \emptyset$ ] do
                For all $u \in \mathcal{N}$ add $t \to u$ and $-u \to -t$ to $C$. [ Lemma 2.1 ]
            Call UpdateS, update all $D_s$'s, $A_s$'s.
Return whichever of $\emptyset$ and $\overline{E}(\mathcal{S}_{\text{act}})$ has a smaller function value.
```

When BSFM-SP took a full step ($\alpha = \lambda_i\beta \le x_{rs}$) it replaced $v^i$ by $v^j$ in $\mathcal{I}$ with the same coefficient. We can translate this directly to BSFM-FC without harming discreteness. Because both $\tilde{x}$ and $\tilde{\lambda}$ are multiplied by SF, this translates to saying that if $\tilde{x}_{rs} \ge \tilde{\lambda}_i\beta$, then we choose $\tilde{\alpha} = \tilde{\lambda}_i\beta$ and take a full step.

When BSFM-SP took a partial step ($\alpha = x_{rs} < \lambda_i\beta$) it replaced $\lambda_i$ by $\lambda_i - x_{rs}/\beta$ and $\lambda_j$ by $x_{rs}/\beta$, which required computing $x_{rs}/\beta$. To keep $\tilde{\lambda}_i$ and $\tilde{\lambda}_j$ integral, we need to compute an integral approximation to $x_{rs}/\beta$. To ensure that the new $\tilde{x}_{rs}$ value hits zero, we need this approximation to be at least as large as $\tilde{x}_{rs}/\beta$.

The natural thing to do is to compute $\tilde{\rho} = \lceil \tilde{x}_{rs}/\beta \rceil$ and update $\lambda_i$ and $\lambda_j$ to $\lambda_i - \tilde{\rho}$ and $\tilde{\rho}$ respectively, which are integers as required. This implies choosing $\tilde{\alpha} = \tilde{\rho}\beta$. Because $\lceil \tilde{x}_{rs}/\beta \rceil < (\tilde{x}_{rs}/\beta) + 1$, $\tilde{\alpha} < \alpha + \beta$. The (at most $\beta$) increase to $\alpha$ above $\tilde{x}_{rs}/\beta$ goes to increasing $\tilde{x}_{sr}$. By Lemma 3.11, $\beta \le \tilde{\delta}$, so we would have that the updated $\tilde{x}_{sr} \le \tilde{\delta}$, so it remains $\tilde{\delta}$-feasible, as desired. Furthermore, we can compute $\tilde{\rho}$ by repeatedly subtracting $\beta$ from $\tilde{x}_{rs}$ until we get a non-positive answer. We started from the assumption that $\tilde{x}_{rs} < \tilde{\lambda}_i\beta$, or $\tilde{x}_{rs}/\beta < \tilde{\lambda}_i$, implying that $\tilde{\rho} \le \tilde{\lambda}_i \le$ SF. Thus the number of subtractions needed is at most SF, which we show below remains small. In fact, we can do better by using repeated doubling: Initialize $q = \beta$ and set $q \leftarrow 2q$ until $q \geq x_{rs}$. The number $d$ of doublings is $O(\log \mathrm{SF}) = O(a)$. Along the way we save $q_i = 2^i q$ for $i = 0, 1, \dots, d$. Then set $q \leftarrow q_{d-1}$, and for $i = d-2, d-3, \dots, 0$, if $q + q_i \leq x_{rs}$ set $q \leftarrow q + q_i$. If the final $q < x_{rs}$, set $q \leftarrow q + \beta$. Thus the final $q$ is of the form $p\beta$ for some integer $p$, we have $q \geq x_{rs}$, and $(p-1)\beta < x_{rs}$. Thus $q = \tilde{\rho}$, and we have computed this in $O(\log \mathrm{SF})$ time.

Due to choosing the initial value of $\tilde{\delta} = 2n\delta^0$ instead of $\delta^0$, we now need to run Fix for $\log_2(4n^3(2n))$ iterations instead of $\log_2(2n^3)$, but this is still $O(\log n)$. This implies that SF stays bounded by a polynomial in $n$, so that the computation of $\tilde{\rho}$ and our simulated multiplications are fully combinatorial operations.

**BSFM-FC Subroutine** Fix$(\hat{f}_t, (\mathcal{S}_{\mathrm{cur}}(t), C), \tilde{\delta})$
Applies to $\hat{f}_t$ defined on closed sets of $(\mathcal{S}_{\mathrm{cur}}(t), C)$, and $\beta \leq \tilde{\delta}$ for all $y \in P_{\mathcal{R}_{\mathrm{cur}}(t)}(\hat{f}_t)$

```
Initialize $\prec$, $\sigma$ as any consistent pair, $\tilde{y} \leftarrow v^{\prec,\sigma}$, SF $\leftarrow 1$, and $\mathcal{N} = \emptyset$.
Initialize $\tilde{x} = \tilde{\varphi} = 0$ and $\tilde{z} = \tilde{y} + \partial^+\tilde{\varphi} + \partial^+\tilde{x}$ ( $= \tilde{y}$).
While SF $\leq 8n^4$ do
    Set SF $\leftarrow$ 2SF, $y \leftarrow 2y$, and $\tilde{\lambda}_i \leftarrow 2\tilde{\lambda}_i$ for $i \in \mathcal{I}$.
    Call BRefine$\mathcal{R}$.
For $u \in \mathcal{S}_{\mathrm{act}}(t)$ do [ add any highly negative nodes to $\mathcal{N}$ ]
    If $\tilde{w}_u < -n^2\tilde{\delta}$ set $\mathcal{N} \leftarrow \mathcal{N} + D_u$.
Return $\mathcal{N}$.
```

From this point the analysis of BSFM-FC proceeds just like the analysis that we did at the beginning of this section.

**Theorem 3.12** *The running time of BSFM-FC is $O(n^9 \mathrm{EO} \log^2 n)$.* ■

## 4 Extensions of BSFM

### 4.1 Finding All BSFM Solutions

In general there can be an exponential number of BSFM solutions. However, it is easy to see that they form a signed ring family, and so there is a compact graph representation of them. This section shows how to construct this representation.

Recall that the complementary slackness conditions for BSFM say that $w$ and $S \in \mathcal{R}$ are jointly optimal if and only if (CS i): $w_e < 0 \Rightarrow e \in S^+$ and $w_e > 0 \Rightarrow e \in S^-$ (which implies that $e \in S^+ \Rightarrow w_e \leq 0$ and $e \in S^- \Rightarrow w_e \geq 0$); and (CS ii): $w(S) = f(S)$ ($S$ is tight). Given any $w$ optimal to BSFM, define index sets $N = \{e \in E \mid w_e < 0\}$, $Z = \{e \in E \mid w_e = 0\}$, and $P = \{e \in E \mid w_e > 0\}$. Then (since any optimal $S$ must be complementary slack to this $w$), to find all optimal BSFM solutions, all we need to do is to find all tight sets for $w$, and then all tight sets $T$ such that $N \subseteq T^+ \subseteq N \cup Z = E - P$ and $P \subseteq T^- \subseteq P \cup Z = E - N$ are precisely all optimal BSFM solutions.

Given that $w = y + \partial^+\varphi$ and the representation of $y$ in (10), note that $T$ is tight for $w$ $\iff$ $\partial^+\varphi(T) = 0$ and $T$ is tight for each $v^i$. Bixby, Cunningham, and Topkis (BCT) [9] give an algorithm for finding the lattice of tight sets for a vertex of a *sub*modular base polyhedron, and Ando and Fujishige [2, Section 3.2] show how to extend this to vertices of bisubmodular polyhedra. For each $i \in \mathcal{I}$ this algorithm requires $O(n^2)$ evaluations of $\hat{f}$, and it produces a directed graph $G^i = (\mathcal{S}_{\text{cur}}, D^i)$ representing the family of tight sets for $v^i$. Now compute the directed graph $G = (\overline{E}, D)$ where $D$ includes all $t \to u$ with $\varphi_{tu} > 0$, and all arcs of each $D^i$. Then any reduced closed set $T$ of $G$ must have $\partial^+\varphi(T) = 0$ and $T$ is tight for each $v^i$, and hence $T$ is tight for $w$. BSFM-SP maintains that $|\mathcal{I}| = O(n)$, and so computing $G$ takes $O(n^3)$ extra function evaluations.

Now contract all strong components of $G$ which are descendents of elements of $P$ into a single node, and those which are ancestors of elements of $N$ into a single node. Every reduced closed set of the partial order of the strong components of the remaining graph induces a tight set for $w$, and also satisfies (CS i), and so is optimal. Conversely, every optimal BSFM solution induces such a reduced closed set. Although there can be an exponential number of BSFM solutions, this contracted graph compactly represents all of them. The material in this section up to this point is an extension from SFM on $2^E$ to BSFM on $\mathcal{R}$ of ideas in Fujishige [22, Remark at the end of Section 14.2], in Murota [38, Note 10.11], and in Nagano [39].

Unfortunately, the various IFF-derived Algorithms, including our BSFM algorithms, do not produce an exact optimal point $w$ (and its representation (10)) required for this algorithm. However, we now show that BSFM-SP and BSFM-FC do carry enough information to get all optimal solutions. As they proceed, they develop sets In and Out such that (by Lemma 3.7) $e \in \text{In} \Rightarrow e$ belongs to *every* BSFM optimal solution, and $e \in \text{Out} \Rightarrow e$ belongs to *no* BSFM optimal solution. They work on the contracted ground set $\mathcal{S}_{\text{act}}$ which excludes elements of In and Out.

They recognize optimality in one of two ways: (1) $|\mathcal{S}_{\text{act}}| \le 1$: In this case the only possible solutions to BSFM on $\mathcal{S}_{\text{cur}}$ are $\emptyset$ and $\mathcal{S}_{\text{act}}$. Therefore the only possible solutions to BSFM on $E$ are In and $\text{In} \cup \bar{E}(\mathcal{S}_{\text{act}})$, and these are easy to check. (2) Scaling parameter $\delta \le 0$: Lemma 3.9 shows that for any consistent $\prec^0$, $\sigma^0$, $w = v^{\prec^0,\sigma^0}$ (with $\varphi \equiv 0$, so that $\partial^+\varphi(T) = 0$ for all $T \in \mathcal{R}_{\text{cur}}$) proves that $\text{abs}(\mathcal{S}_{\text{cur}})|\sigma^0$ solves BSFM. Therefore we can apply the algorithm of [2] to find all tight sets w.r.t. $w$, which then gives us all BSFM solutions to the original problem. This shows that to get all optimal BSFM solutions, we do not actually need an exact $w$ defined over all of $E$. It is enough that BSFM-SP (and BSFM-FC) supply an exact optimal $w$ defined only on $\text{abs}(\mathcal{S}_{\text{cur}})$.

## 4.2 Minimizing Separable Convex Objectives on $P_{\mathcal{R}}(f)$

Fujishige [20, Section 5] gives an algorithmic framework for minimizing $\sum_e g_e(w_e)$ s.t. $w \in P_{\mathcal{R}}(f)$, where each $g_e$ is convex. This algorithm assumes two things: (1) We can compute (or are given) $\bar{w}_e = \operatorname{argmin} g_e(w_e)$ if it exists. Define $\bar{w}_e = +\infty$ if $g_e(w_e)$ is monotone decreasing, and $\bar{w}_e = -\infty$ if $g_e(w_e)$ is monotone increasing. Note that in some cases computing $\bar{w}$ might already not be do-able in strongly polynomial time: Hochbaum [29] shows that even square roots cannot be computed in strongly polynomial time. However, in practice computing $\bar{w}$ is usually easy. (2) An oracle for computing *signed exchange capacities*: For $w \in P_{\mathcal{R}}(f)$ and $e \in E$, define $c(w, \pm e) = \min\{f(S) - w(S) \mid e \in S^{\pm},\ S \in \mathcal{R}\}$, and for $e \ne g \in E$ define $c(w, \pm e, +g) = \min\{f(S) - w(S) \mid (e \in S^{\pm} \text{ and } g \notin S^+ \cup S^-) \text{ or } (e \notin S^+ \cup S^- \text{ and } g \in S^+)\}$ and $c(w, \pm e, -g) = \min\{f(S) - w(S) \mid (e \in S^{\pm} \text{ and } g \notin S^+ \cup S^-) \text{ or } (e \notin S^+ \cup S^- \text{ and } g \in S^-)\}$. Note that any of these can be computed with one call to a BSFM algorithm (given that the $w$

appearing here is likely to be fractional, BSFM-SP is likely to be helpful).

The algorithm proceeds as follows: Let $w^0 \in P_{\mathcal{R}}(f)$ be some initial feasible point, perhaps coming from Signed Greedy on a consistent pair. Then the algorithm moves $w^0$ towards $\bar{w}$ in $P_{\mathcal{R}}(f)$ by computing $O(n^2)$ signed exchange capacities to compute an optimal solution (the algorithm is able to recognize when the solution is unbounded). Putting together our Theorems 3.10 and 3.12, and [20, Theorems 4.1, 5.1] we get:

**Theorem 4.1** *If we can compute $\bar{w}$, then we can minimize separable convex objectives over $P_{\mathcal{R}}(f)$ in $O(n^9 \mathrm{EO} \log n)$ strongly polynomial, and $O(n^{11} \mathrm{EO} \log^2 n)$ fully combinatorial time.*

■

### 4.3 Solving a Line Search Problem on $P_{\mathcal{R}}(f)$

The signed exchange capacities of the previous section are special cases of the more general *Line Search* problem: Given some $w \in P_{\mathcal{R}}(f)$, and a *direction vector* $a \in \mathbb{R}^E$, compute $\max\{\alpha \mid w + \alpha a \in P_{\mathcal{R}}(f)\}$. Nagano [39] shows how to solve this problem for base polyhedra and submodular polyhedra coming from submodular functions. Here we adapt Nagano's ideas for the bisubmodular case.

Nagano's algorithm uses Megiddo's [37] general framework for parametric optimization, which requires a "linear" algorithm for solving the underlying non-parametric algorithm, i.e., an algorithm that generates only linear functions of its data when making comparisons. Among the SFM algorithms, only I-FC is linear in this sense, and so Nagano embeds I-FC inside Megiddo's framework to get a strongly polynomial algorithm for the SFM case.

By using our BSFM-FC algorithm we can replicate this idea. Note that the optimal solution is $+\infty$ iff $a$ is an unbounded direction for $P_{\mathcal{R}}(f)$ iff there is some flow $x \geq 0$ on $(\overline{E}, C)$ with $\delta^+ x = a$. This can be checked using the skew flow methods of Goldberg and Karzanov [25].

## 5 Conclusions and Open Questions

This paper finds the first combinatorial strongly polynomial and fully combinatorial algorithms for BSFM. Given that many of the applications of BSFM are to separation problems, which potentially could involve highly fractional points, it is worthwhile to have a combinatorial strongly polynomial algorithm.

This work also deepens our understanding of the technical issues involved in trying to extend IFF algorithmic techniques to other problems. On one hand it is encouraging that we were able to extend as far as we did, but on the other hand it is discouraging that we (so far) were not able to extend the Hybrid Algorithm [31] from SFM to BSFM.

Recall that Hybrid gets its speed-up from modifying blocks of consecutive components of its $\prec^i$'s, instead of just consecutive pairs. By choosing these blocks w.r.t. distance labels, Hybrid's HRefine is $O(n)$ faster than Refine. It is not too hard to figure out how to do similar block changes that preserve consistency in the BSFM context. The hurdle that we have not been able to overcome is in how the distance labels interact with Tail-Exchange: it is not clear how to define labels that properly cause bad elements to first move right, then move left after getting their signs flipped.

Hence a big open question is whether Hybrid can be extended to BSFM. We can ask the same question about whether Schrijver's Algorithm (the original version [48, 51] or the Push-Relabel version [18]), or Orlin's Algorithm [42] be extended to BSFM? The most intriguing at this point is Orlin's algorithm, as it is $O(n \log n)$ faster than the strongly polynomial version of Hybrid, which had been the fastest among combinatorial algorithms.

All of the combinatorial SFM and BSFM algorithms use the device of representing the current feasible point as a convex combination of vertices as in (10). This device contributes to the complexity of the algorithms, is rather inelegant to implement, and detracts from the esthetics of the algorithms. It is not clear that SFM or BSFM algorithms essentially need to use linear algebra in this way. Hence it would be nice to find SFM or BSFM algorithms that use some other method to represent their current points. One possibility is the "combinatorial hull", see [21].

Section 4.1 raises the following question. It is easy to see that in fact for both SFM and BSFM, with integral data there always exists an *integral* optimal solution $w$, but apparently none of the existing algorithms can directly find one. It is possible to use $O(n)$ calls to an SFM or BSFM algorithm to find an integral optimal point, but it is not clear how to keep or compute representation (10) of its bounded part (which was quite useful in Section 4.1). Thus it would be interesting to find a BSFM (or even SFM) algorithm that can find an integral optimal point $w$ together with a representation of its bounded part as a convex combination of vertices.

Both SFM and BSFM are problems where a simple greedy algorithm optimizes over the underlying polyhedron, but where the problem becomes much more difficult with even a very simple separable piece-wise linear convex objective. Nagano [40] shows how to adapt any parametric SFM algorithm to solve general separable convex objectives over the base polytope of a submodular function. Section 4.2 shows how our algorithms plus the framework of [20] suffice to get a combinatorial polynomial algorithm for minimizing separable convex objectives over $P_{\mathcal{R}}(f)$, but it would be interesting to get a more direct and faster algorithm for this problem.

Another problem that fits into the "linear easy, separable convex not so easy" paradigm is optimizing over zonotopes (see [23]). Linear optimization is trivial, but finding a minimum $\ell_2$-norm point is as hard as linear programming, and there is some hope that this approach could lead to a strongly polynomial algorithm for linear programming. Perhaps other such problems exist.

Finally, we have placed this paper in the context of an overall stream of research aimed at finding combinatorial polynomial algorithms for problems whose only known polynomial algorithm is Ellipsoid-based. Hence it would be very nice to be able to broaden the applications of the algorithmic techniques herein to other problems. One possible target would be to solve the separation problem over the polytope of subtour elimination constraints for the Traveling Salesman Problem (see, e.g., [6, Chapter 6]). Another possible target from the TDI point of view would be Schrijver's general framework for TDI [46].

**Acknowledgment**

We thank Satoru Iwata for his substantial help with this material.

## References

[1] K. Ando and S. Fujishige (1994). $\sqcup$, $\sqcap$-Closed Families and Signed Posets. Report no. 93813, Forschunginstitut für Diskrete Mathematik, Universität Bonn.

[2] K. Ando and S. Fujishige (1996). On Structures of Bisubmodular Polyhedra. *Math. Prog.*, **74**, 293–317.

[3] K. Ando, S. Fujishige and T. Naitoh (1996). A Characterization of Bisubmodular Functions. *Discrete Mathematics*, **148**, 299–303.

[4] K. Ando, S. Fujishige and T. Nemoto (1996). The Minimum-Weight Ideal Problem for Signed Posets. *J. OR Soc. of Japan*, **39**, 558–565.

[5] K. Ando, S. Fujishige and T. Nemoto (1996). Decomposition of a Bidirected Graph into Strongly Connected Components and its Signed Poset Structure. *Disc. Appl. Math.*, **68**, 237–248.

[6] D.L. Applegate, R.E. Bixby, V. Chvátal and W.J. Cook (2007). *The Traveling Salesman Problem: A Computational Study.* Princeton University Press, Princeton, NJ.

[7] J.M. Bilbao Arrese, J.R. Fernández García, M. Jiménez Jiménez, J.J. López Vázquez (2005). A Survey of Bicooperative Games. *Proceedings of the 4th Twente Workshop on Cooperative Game Theory*, Enschede, the Netherlands, 5–15.

[8] G. Birkhoff (1967). *Lattice Theory.* Amer. Math. Soc..

[9] R. E. Bixby, W. H. Cunningham, and D. M. Topkis (1985). The Partial Order of a Polymatroid Extreme Point. *Math. of OR*, **10**, 367–378.

[10] A. Bouchet (1989). Matchings and $\Delta$-Matroids. *Discrete Mathematics*, **24**, 55–62.

[11] A. Bouchet and W.H. Cunningham (1995). Delta-Matroids, Jump Systems and Bisubmodular Games. *SIAM J. on Disc. Math.*, **8**, 17–32.

[12] R. Chandrasekaran and S.N. Kabadi (1988). Pseudomatroids. *Disc. Math.*, **71**, 205–217.

[13] W. H. Cunningham (1984). Testing Membership in Matroid Polyhedra. *JCT Series B*, **36**, 161–188.

[14] W.H. Cunningham and J. Green-Krótki (1991). *b*-Matching Degree Sequence Polyhedra. *Combinatorica*, **11**, 219–230.

[15] W.H. Cunningham and J. Green-Krótki (1994). A Separation Algorithm for the Matchable Set Polytope. *Math. Prog.*, **65**, 139–150.

[16] A. Dress and T.F. Havel (1986). Some Combinatorial Properties of Discriminants in Metric Vector Spaces. *Adv. Math.*, *62*, 285–312.

[17] F.D.J. Dunstan and D.J.A. Welsh (1973). A Greedy Algorithm for Solving a Certain Class of Linear Programmes. *Math. Prog.*, **62**, 338–353.

[18] L. K. Fleischer and S. Iwata (2003). A Push-Relabel Framework for Submodular Function Minimization and Applications to Parametric Optimization. "Submodularity" special issue of *Discrete Applied Mathematics*, S. Fujishige ed., **131**, 311-322.

[19] S. Fujishige (1984). A System of Linear Inequalities with a Submodular Function on $\{0, \pm 1\}$ Vectors. *Lin. Alg. and its Appl.*, **63**, 253–266.

[20] S. Fujishige (1997). A Min-Max Theorem for Bisubmodular Polyhedra. *SIAM J. Disc. Math*, **10**, 294–308.

[21] S. Fujishige (2003). Submodular Function Minimization and Related Topics. *Optimization Methods and Software*, **18**, 169–180.

[22] S. Fujishige (2005). *Submodular Functions and Optimization.* Second Edition. North-Holland.

[23] S. Fujishige, T. Hayashi, and S. Isotani (2006). The Minimum-Norm-Point Algorithm Applied to Submodular Function Minimization and Linear Programming. Research Institute for Mathematical Sciences Preprint RIMS-1571, Kyoto University, Kyoto Japan.

[24] S. Fujishige and S. Iwata (2006). Bisubmodular Function Minimization. *SIAM J. Disc. Math.*, **19**, 1065–1073; an earlier version appeared in *Proceedings of the 8th Conference on Integer Programming and Combinatorial Optimization (IPCO Utrecht 2001)*, Lecture Notes in Computer Science 2081, K. Aardal and B. Gerards eds., Springer, Berlin, 160–169.

[25] A. V. Goldberg and A. V. Karzanov (2004). Maximum Skew-Symmetric Flows and Matchings. *Math. Programming*, **100**, 537–568.

[26] A. V. Goldberg and R. E. Tarjan (1990). Finding Minimum-Cost Circulations by Successive Approximation. *Mathematics of Operations Research*, **15**, 430–466.

[27] M. Grötschel, L. Lovász, and A. Schrijver (1981). The Ellipsoid Algorithm and its Consequences in Combinatorial Optimization. *Combinatorica*, **1**, 499–513.

[28] M. Grötschel, L. Lovász, and A. Schrijver (1988). *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag.

[29] D. S. Hochbaum (1993). Polynomial and strongly polynomial algorithms for convex network optimization. In *Network Optimization Problems*, D. Z. Du and P. M. Pardalos, eds., pp. 63–92.

[30] S. Iwata (2002). A Fully Combinatorial Algorithm for Submodular Function Minimization. *J. Combin. Theory Ser. B*, **84**, 203–212.

[31] S. Iwata (2003). A Faster Scaling Algorithm for Minimizing Submodular Functions. *SIAM J. on Computing*, **32**, 833–840; an extended abstract appeared in *Proceedings of the 9th Conference on Integer Programming and Combinatorial Optimization (IPCO MIT)*, Lecture Notes in Computer Science 2337, W. J. Cook and A. S. Schulz eds., Springer, Berlin, 1–8.

[32] S. Iwata (2008). Submodular Function Minimization. *Mathematical Programming*, **112**, 45–64.

[33] S. Iwata, L. Fleischer, and S. Fujishige (2001). A Combinatorial, Strongly Polynomial-Time Algorithm for Minimizing Submodular Functions. *J. ACM*, **48**, 761–777.

[34] S.N. Kabadi (1984). Characterization and Development of Solution Methods for Special Classes of Totally Dual Integral Systems. Dissertation, School of Management, University of Texas at Dallas.

[35] S.N. Kabadi and R. Chandrasekaran (1990). On Totally Dual Integral Systems. *Disc. Appl. Math.*, **26**, 87–104.

[36] S. T. McCormick (2006). Submodular Function Minimization. A chapter in the *Handbook on Discrete Optimization*, Elsevier, K. Aardal, G. Nemhauser, and R. Weismantel, eds.

[37] Megiddo, N. (1979). Combinatorial Optimization With Rational Objective Functions. *Math. Oper. Res.* **4**; 414–424.

[38] K. Murota (2003). *Discrete Convex Analysis.* SIAM Monographs on Discrete Mathematics and Applications, Society for Industrial and Applied Mathematics, Philadelphia.

[39] K. Nagano (2005). A Strongly Polynomial Algorithm for Line Search in Submodular Polyhedra. *Proceedings of the 4th Japanese-Hungarian Symposium on Discrete Mathematics and Its Applications*, Budapest, Hungary, 234–242.

[40] K. Nagano (2007). On Convex Minimization over Base Polytopes. *Proceedings of IPCO 12*, M. Fischetti and D. Williamson, eds., Ithaca, NY, 252–266.

[41] M. Nakamura (1988). A Characterization of Greedy Sets — Universal Polymatroids I. *Scientific Papers of the College of Arts and Sciences, The University of Tokyo*, **38**, 1555–167.

[42] J.B. Orlin (2007). A Faster Strongly Polynomial Algorithm for Submodular Function Minimization. *Proceedings of IPCO 12*, M. Fischetti and D. Williamson, eds., Ithaca, NY, 240–251.

[43] L. Qi (1988). Directed Submodularity, Ditroids, and Directed Submodular Flows. *Math. Prog.*, **42**, 579–599.

[44] M.N. Queyranne and Y. Wang (1991). Single-Machine Scheduling Polyhedra with Precedence Constraints. *Math. of OR*, **16**, 1–20.

[45] V. Reiner (1993). Signed Posets. *J. Comb. Theory A*, **62**, 324–360.

[46] Schrijver, A. (1984). Proving Total Dual Integrality with Cross-free Families — A General Framework. *Mathematical Programming* **29**, 15–27.

[47] A. Schrijver (1986). *Theory of Linear and Integer Programming.* John Wiley & Sons, New York, NY.

[48] A. Schrijver (2000). A Combinatorial Algorithm Minimizing Submodular Functions in Strongly Polynomial Time. *J. Combin. Theory Ser. B* **80**, 346–355.

[49] A. Schrijver (2003). *Combinatorial Optimization: Polyhedra and Efficiency.* Springer, Berlin.

[50] É. Tardos (1985). A Strongly Polynomial Minimum Cost Circulation Algorithm. *Combinatorica*, **5**, 247–256.

[51] J. Vygen (2003). A Note on Schrijver's Submodular Function Minimization Algorithm. *JCT B*, **88**, 399–402.

[52] F. Zhang (2003). A Separation Algorithm for $b$-Matching Degree-Sequence Polyhedra. *Math. Oper. Res.*, **28**, 92-102.