Preprint 99 – 043

Maximum Skew-Symmetric Flows and Their Applications to B-Matchings

by

Andrew V. Goldberg
and
Alexander V. Karzanov

Mitgeteilt von W. Deuber
7.4.1999

Maximum Skew-Symmetric Flows and Their Applications to B-Matchings

Andrew V. Goldberg *

InterTrust Technologies Corp.
460 Oakmead Parkway
Sunnyvale, CA 94086
email: *goldberg@intertrust.com*

Alexander V. Karzanov †

Institute for System Analysis
9, Prospect 60 Let Oktyabrya
117312 Moscow, Russia
email: *sasha@cs.isa.ac.ru*

March 1999

Abstract

We introduce the maximum integer skew-symmetric flow problem (MSFP) which generalizes both the maximum flow and maximum matching problems. We establish analogs of the classical flow decomposition, augmenting path, and max-flow min-cut theorems for skew-symmetric flows. These theoretical results are then used to develop an $O(M(n,m) + nm)$ time algorithm for solving the MSFP, where $M(n,m)$ is the time needed to find a maximum integer (usual) flow in a network with n nodes and m arcs. This gives a method of the same complexity for the capacitated b-matching problem. Other methods for solving the MSFP are also discussed.

*Part of this work was done while the author was at NEC Research Institute, Inc., Princeton, NJ.

†Supported in part by RFBR grant 97-01-00115 and a grant from the Sonderforshungsbereich 343, Bielefeld Universität, Bielefeld.

1 Introduction

This paper continues a study of combinatorial and algorithmic properties of problems on skew-symmetric graphs, and their applications, started in our previous paper [14]. That paper, devoted to *regular path* problems, extends the usual reachability and shortest path problems to skew-symmetric graphs. The present paper deals with the *maximum integer skew-symmetric flow* (or *maximum IS-flow*) problem, abbreviated as *MSFP*. We study combinatorial and linear programming properties of this problem and develop fast algorithms for it.

That the bipartite matching problem can be viewed as a special case of the maximum flow problem is well-known [9]. The combinatorial structure of non-bipartite matchings is, however, somewhat more complicated than the structure of flows (cf. [5, 20]). This phenomenon explains, to some extent, why general matching algorithms are typically more intricate relative to flow algorithms. The maximum IS-flow problem is a generalization of both the maximum flow and maximum matching problems. Moreover, this generalization appears to be well-grounded for two reasons. First, the basic combinatorial and linear programming theorems for usual flows have quite appealing counterparts for IS-flows. Second, when solving problems on IS-flows, we are able to use some intuition, ideas and technical tools well-understood for usual flows so that the implied algorithms for matchings become more comprehensible.

As the maximum flow problem is related to certain path problems, the maximum IS-flow problem is related to regular path problems in skew-symmetric graphs. We extensively use theoretical and algorithmic results of [14] in this paper.

The mini-theory for IS-flows that we develop is parallel to that for usual maximum flows [9]. The basic results of the flow theory are the decomposition, augmenting path, and max-flow min-cut theorems. The flow decomposition theorem says that a flow can be decomposed into a collection of source-to-sink paths and cycles; the IS-flow decomposition theorem says that an IS-flow can be decomposed into a collection of pairs of symmetric source-to-sink paths and pairs of symmetric cycles. The augmenting path theorem says that a flow is maximum if and only if it admits no augmenting path. The IS-flow augmenting path theorem says that an IS-flow is maximum if and only if it admits no regular augmenting path. The max-flow min-cut theorem says that the maximum flow value is equal to the minimum cut capacity. Its skew-symmetric equivalent is that the maximum IS-flow value is equal to the minimum *odd-barrier* capacity.

Linear programming duality plays an important role in understanding the structure of network flow problems and in developing algorithms for these problems. We discuss a linear programming description of the maximum IS-flow problem and the complementary slackness conditions for it.

Based on theoretical results, we then develop an $O(M(n,m) + nm)$-time algorithm for the MSFP that uses a max-flow algorithm to construct a good initial solution and then improves it by augmentations along regular paths found by using a regular reachability algorithm with linear complexity. Here and later on n and m denote the numbers of nodes and arcs of the input network, respectively, and $M(n,m)$ is the time needed to find an integer maximum flow in a network with n nodes and m arcs (see [15] for the current state of the art in maximum flow algorithms). As a consequence, we obtain an algorithm with the same complexity for the maximum and feasible capacitated b-matching problems [6, 20], due to a simple reduction of these to the MSFP.

Next, well-known polynomial methods for the max-flow problem are the shortest augmenting path algorithm of Edmonds and Karp [7], and its improved version, the blocking flow algorithm of Dinitz [4]. We show that the ideas of both methods can be generalized for IS-flows. This results in our shortest regular augmenting path and blocking IS-flow methods for the MSFP. These methods extensively handle dual objects arising when the shortest regular path algorithm from [14] is applied to the residual graphs. We show that, when making flow augmentations along shortest regular augmenting paths, the length of such paths is monotonely non-decreasing. Moreover, the MSFP is reduced to finding $O(n)$ blocking IS-flows in acyclic networks with at most n nodes and m arcs each. In the cases of unit arc capacities and unit "node capacities" the number of blocking IS-flows that we need to construct becomes $O(m^{1/2})$ and $O(n^{1/2})$, respectively, similarly to that for the corresponding usual networks [8, 17, 18].

This paper is organized as follows. Section 2 gives basic definitions. Sections 3 and 4 are devoted to theoretical results, considering combinatorial and linear programming aspects of IS-flows, respectively. Section 5 describes the algorithm for the MSFP based on finding a good initial solution. The shortest regular augmenting path algorithm and a high level description of the blocking IS-flow method are given in Section 6. Necessary facts about the regular path problems and their solution methods are reviewed in Section 7. Using these, Section 8 specifies the subproblem arising at iterations of the blocking IS-flow method. It also estimates the number of iterations for special skew-symmetric networks. Section 9 demonstrates applications to b-matchings, and Section 10 contains concluding remarks.

2 Preliminaries

A digraph $G = (V, E)$ is called *skew-symmetric* if there is a mapping σ of $V \cup E$ onto itself such that: σ is an involution (*i.e.*, $\sigma(x) \neq x$ and $\sigma(\sigma(x)) = x$ for any $x \in V \cup E$); for every $v \in V$, $\sigma(v) \in V$; and for every $a = (v, w) \in E$, $\sigma(a) = (\sigma(w), \sigma(v))$. (Although parallel arcs are allowed in G, when it is not confusing, an arc leaving a node x and entering a node y is denoted by (x, y).) We usually assume that the description of G includes σ. For brevity, we often use the term *symmetric* instead of skew-symmetric. We say that the node (arc) $\sigma(x)$ is symmetric to a node (arc) x. Symmetric elements are also called *mates*, and we use notation with primes for mates, denoting by x' the mate $\sigma(x)$ of an element x. Note that G can contain an arc a from a node v to its mate v'; then a' is also an arc from v to v'.

Unless mentioned otherwise, when talking about paths (cycles), we mean directed paths (cycles). The symmetry σ is extended in a natural way to paths, subgraphs, and other objects in G; e.g., two paths (cycles) are symmetric if the elements of one of them are symmetric to those of the other and go in the reverse order. Note that G cannot contain self-symmetric paths or cycles. Indeed, if $P = (x_0, a_1, x_1, \dots, a_k, x_k)$ is such a path (cycle), choose arcs a_i and a_j such that $i \leq j$, $a_j = \sigma(a_i)$ and $j - i$ is minimum. Then $j > i + 1$ (as $j = i$ would imply $\sigma(a_i) = a_i$ and $j = i + 1$ would imply $\sigma(x_i) = x_{j-1} = x_i$). Now $\sigma(a_{i+1}) = a_{j-1}$ contradicts the minimality of $j - i$.

A function h on E is said to obey the *symmetry condition* if $h(a) = h(a')$ for all $a \in E$. Throughout, by a *symmetric* function we always mean a *nonnegative integer-valued* function on the arcs of G (or another skew-symmetric graph in question) which satisfies the symmetry condition.

A *skew-symmetric network* is a quadruple $N = (G, \sigma, u, s)$ consisting of a skew-symmetric graph $G = (V, E)$ with symmetry σ, a symmetric function u (of *arc capacities*) on E, and a *source* $s \in V$. The mate s' of s is considered as the *sink* of N. A *flow* in N is a function $f : E \to \mathbf{R}_+$ satisfying the capacity constraints

$$f(a) \leq u(a) \qquad \text{for all } a \in E$$

and the conservation constraints

$$\text{div}_f(x) := \sum_{(x,y)\in E} f(x,y) - \sum_{(y,x)\in E} f(y,x) = 0 \qquad \text{for all } x \in V - \{s, s'\}.$$

The value $\text{div}_f(s)$ is called the *value* of f and denoted by $|f|$; we usually assume that $|f| \geq 0$. An *IS-flow* abbreviates a *symmetric integer flow*, the main object that we study in this paper. The *maximum skew-symmetric flow problem (MSFP)* is to find an IS-flow of maximum value in N.

The integrality requirement is important: if we do not require f to be integral, then for any integer flow f in N, the flow f', defined by $f'(a) = (f(a) + f(a'))/2$ for $a \in E$, is a maximum flow satisfying the symmetry condition but being not necessarily integral.

Note that, given a digraph $D = (V(D), A(D))$ with two specified nodes p and q and nonnegative integer capacities of the arcs, we can construct a skew-symmetric graph G by taking a disjoint copy D' of D with all arcs reversed, adding two extra nodes s and s', and adding four arcs $(s,p), (s,q'), (q,s'), (p',s')$ of infinite capacity, where p', q' are the copies of p, q in D', respectively. Then the integer flows from p to q in D one-to-one correspond, in a natural way, to the IS-flows from s to s' in G. This shows that the MSFP generalizes the classical (integer) max-flow problem. Also the MSFP generalizes corresponding problems on matchings and b-matchings, as we explain in Section 9.

In our study of IS-flows we will rely on results on regular paths in skew-symmetric graphs. A *regular path* (or an *r-path*) is a path in G that does not contain a pair of symmetric arcs. Similarly, an *r-cycle* is a cycle that does not contain a pair of symmetric arcs. The *r-reachability problem (RP)* is to find an r-path from s to s' or a proof that there is none. Given a symmetric function of *arc lengths*, the *shortest r-path problem (SPP)* is to find a minimum length r-path from s to s' or a proof that there is none.

A criterion for the existence of a regular s–s' path is less trivial than that for the usual path reachability; it involves so-called barriers. We say that

$$\mathcal{B} = (A; X_1, \ldots, X_k)$$

is an *s-barrier* if the following conditions hold.

(B1) $A, X_1, \ldots, X_k$ are pairwise disjoint subsets of V, and $s \in A$.

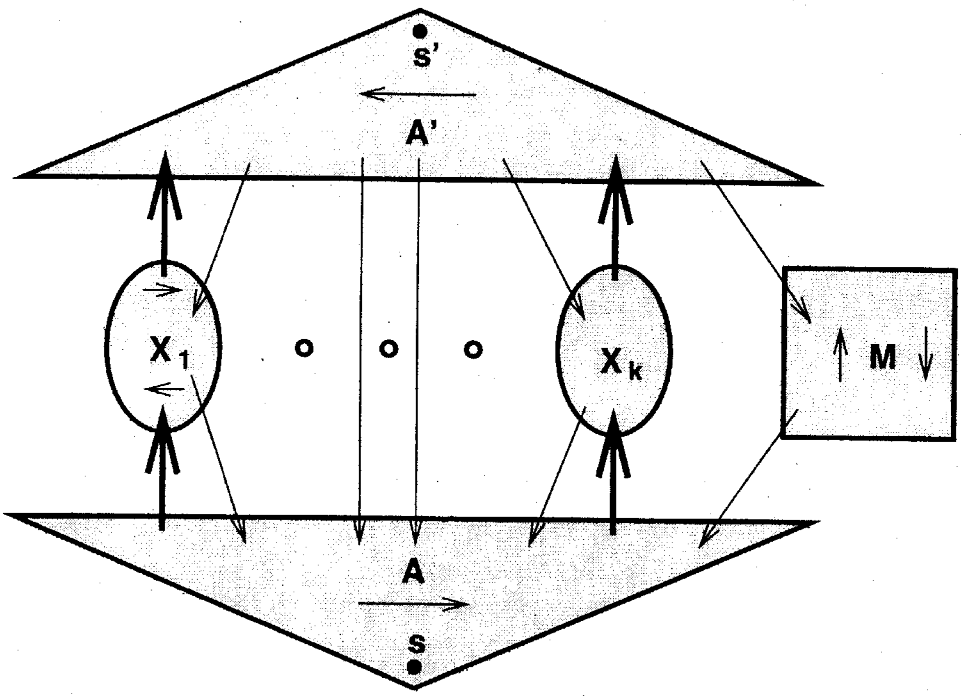


Figure 1: *A barrier.*

(B2) For $A' = \sigma(A)$, $A \cap A' = \emptyset$.

(B3) For $i = 1, \ldots, k$, X_i is symmetric, *i.e.*, $\sigma(X_i) = X_i$.

(B4) For $i = 1, \ldots, k$, there is a unique arc, e^i, from A to X_i.

(B5) For $i, j = 1, \ldots, k$ and $i \neq j$, no arc connects X_i and X_j.

(B6) For $M = V - (A \cup A' \cup X_1 \cup \ldots \cup X_k)$ and $i = 1, \ldots, k$, no arc connects X_i and M.

(B7) No arc goes from A to $A' \cup M$.

(Note that arcs from A' to A, from X_i to A, and from M to A are possible.) Figure 1 illustrates the definition.

Theorem 2.1 [14] *There is an r-path from s to s' if and only if there is no s-barrier.*

This criterion will be used in Section 3 to obtain an analog of the max-flow min-cut theorem for IS-flows.

Theorem 2.2 [2, 14] *The r-reachability problem in G can be solved in O(m) time.*

The methods for the maximum IS-flow problem that we develop apply, as a subroutine, the r-reachability algorithm of linear complexity from [14] which finds either a regular s–s' path or an s-barrier. Another ingredient used in our methods is the shortest r-path algorithm for the case of nonnegative symmetric lengths, which runs in $O(m \log n)$ time, and in $O(m)$ time for all-unit lengths [14]. The needed results concerning the RP and SPP are discussed in more details in Section 7.

In the rest of this paper, σ and s will denote the symmetry map and the source, respectively, regardless of the network in question, which will allow us to use the shorter notation (G, u) for a network (G, σ, u, s). Given a simple path P, the number of arcs on P is denoted by $|P|$ and the incidence vector of its arc set in $\mathbf{R}^E$ is denoted by χ^P, i.e., $\chi^P(a) = 1$ if a is an arc of P, and 0 otherwise.

3 Skew-Symmetric Flow Theory

In this section we extend the classical flow decomposition, augmenting path, and max-flow min-cut theorems of Ford and Fulkerson [9] to the skew-symmetric case.

Let h be a symmetric function on the arcs of a skew-symmetric graph $G = (V, E)$, and let $\text{supp}(h)$ denote its *support* $\{a \in E : h(a) > 0\}$. A path (cycle) P in G is called *h-regular* if all arcs of P belong to $\text{supp}(h)$ and each arc $a \in P$ such that $a' \in P$ satisfies $h(a) \geq 2$. Clearly when h is all-unit on E, the sets of regular and h-regular paths (cycles) are the same. We call an arc a of P *ordinary* if $a' \notin P$ and define the *h-capacity* $\delta_h(P)$ of P to be the minimum of numbers $h(a)$ for ordinary arcs a and numbers $\lfloor h(a)/2 \rfloor$ for non-ordinary arcs a in P.

To state the symmetric flow decomposition theorem, consider an IS-flow f in a skew-symmetric network $N = (G = (V, E), u)$. An IS-flow g in N is called *elementary* if it is representable as $g = \delta\chi^P + \delta\chi^{P'}$, where P is a simple cycle or a simple path from s to s' or a simple path from s' to s, $P' = \sigma(P)$, and δ is a *positive* integer. Since g is feasible, P is u-regular and $\delta \leq \delta_u(P)$. We also denote g as (P, P', δ). By a *symmetric decomposition* of f we mean a set D of elementary flows such that $f = \sum(g : g \in D)$. The *symmetric decomposition theorem* is the following.

Theorem 3.1 *For an IS-flow f in G, there exists a symmetric decomposition consisting of at most m elementary flows.*

Proof. We may assume that f is nonzero. We build up an f-regular path Γ in G until this path contains a simple cycle P or a simple path P connecting s and s'. This will determine a member of the desired flow decomposition. Then we accordingly decrease f and repeat the process for the resulting IS-flow f', and so on.

We start with Γ formed by a single arc $a \in \text{supp}(f)$. First we grow Γ forward. Let $b = (v, w)$ be the last arc on Γ. Suppose that $w \neq s, s'$. By the conservation for f at w, $\text{supp}(f)$ must contain an arc $q = (w, z)$. If q' is not on Γ or $f(q) \geq 2$, we add q to Γ.

Suppose q' is on Γ and $f(q) = 1$. Let Γ_1 be the part of Γ between w' and w. Then Γ_1 contains at least one arc since $w \neq w'$. Suppose there is an arc $\tilde{q} \in \text{supp}(f)$ leaving w and different from q.

Then we can add $\tilde{q}$ to Γ instead of q, forming a longer f-regular path. Now suppose that such a $\tilde{q}$ does not exist. Then exactly one unit of the flow f leaves w. Hence, exactly one unit of the flow f enters w, implying that b as above is the only arc entering w in supp(f), and that $f(b) = 1$. But $\sigma(d)$ also enters w, where d is the first arc on Γ_1. The fact that $\sigma(d) \neq b$ (since Γ_1 is f-regular) leads to a contradiction.

Let (w, z) be the arc added to Γ. If z is not on Γ, then Γ is a simple f-regular path, and we continue growing Γ. If z is on Γ, we discover a simple f-regular cycle P.

If Γ reaches s' or s, we start growing Γ backward from the initial arc a in a way similar to growing it forward. We stop when an f-regular cycle P is found or one of s, s' is reached. In the latter case $P = \Gamma$ is either an f-regular path from s to s' or from s' to s, or an f-regular cycle (containing s or s').

Form the elementary flow $g = (P, P', \delta)$ with $\delta = \delta_f(P)$ and reduce f to $f' = f - \delta\chi^P - \delta\chi^{P'}$. Since P is f-regular, $\delta > 0$. Moreover, there is a pair e, e' of symmetric arcs of P such that either $f'(e) = f'(e') = 0$ or $f'(e) = f'(e') = 1$; we associate such a pair with g. In the former case e, e' vanish in the support of the new IS-flow f', while in the latter case e, e' can be used in further iterations of the decomposition process at most once. Therefore, each pair of arc mates of G is associated with at most two members of the constructed decomposition D, yielding $|D| \leq m$. ■

The above proof gives a polynomial time algorithm for symmetric decomposition. Moreover, the above decomposition process can be easily implemented in $O(nm)$ time, which matches the complexity of standard decomposition algorithms for usual flows.

The decomposition theorem and the fact that the network has no self-symmetric paths imply the following useful property.

Corollary 3.2 *For any symmetric set $S \subseteq V$ and any IS-flow in G, the total flow on the arcs entering S, as well as the total flow on the arcs leaving S, is even.*

Remark. Another consequence of Theorem 3.1 is that w.l.o.g. we may assume that no arc of G enters s. Indeed, consider a maximum IS-flow f in G and a symmetric decomposition D of f. Putting together the elementary flows from s to s' in D, we obtain an IS-flow f' in G with $|f'| \geq |f|$, so f' is a maximum flow. Since f' uses no arc entering s or leaving s', deletion of all such arcs from G produces an equivalent problem in a skew-symmertric graph.

Next we state a symmetric version of the augmenting path theorem. It is convenient to consider the graph $G^+ = (V, E^+)$ formed by adding a reverse arc (y, x) to each arc (x, y) of G. For $a \in E^+$, a^R denotes the corresponding reverse arc. The symmetry σ is extended to E^+ in a natural way. Given a symmetric capacity function u on E and an IS-flow f on G, define the *residual capacity* $u_f(a)$ of an arc $a \in E^+$ to be $u(a) - f(a)$ if $a \in E$, and $f(a^R)$ otherwise. An arc $a \in E^+$ is called *residual* if $u_f(a) > 0$, and *saturated* otherwise. Given an IS-flow g in the network (G^+, u_f), we define the function $f \oplus g$ on E by setting $(f \oplus g)(a) = f(a) + g(a) - g(a^R)$. Clearly $f \oplus g$ is a feasible IS-flow in (G, u) whose value amounts to $|f| + |g|$.

By an *r-augmenting path* for f we mean a u_f-regular path from s to s' in G^+. If P is an r-augmenting path and if $\delta \in \mathbf{N}$ does not exceed the u_f-capacity of P, then we can push δ units of flow through (not necessarily directed) path in G corresponding to P and then δ units through the path corresponding to P'. Formally, f is transformed into $f \oplus g$, where g is the elementary flow (P, P', δ) in (G^+, u_f). Such an augmentation increases the value of f by 2δ.

Theorem 3.3 *An IS-flow f is maximum if and only if there is no r-augmenting path.*

Proof. The direction that the existence of an r-augmenting path implies that f is not maximum is obvious in light of the above discussion.

To see the other direction, suppose that f is not maximum, and let f^* be a maximum IS-flow in G. For $a \in E$ define $g(a) = f^*(a) - f(a)$ and $g(a^R) = 0$ if $f^*(a) \geq f(a)$, while $g(a^R) = f(a) - f^*(a)$ and $g(a) = 0$ if $f^*(a) < f(a)$. One can see that g is a feasible symmetric flow in (G^+, u_f). Take a symmetric decomposition D of g. Since $|g| = |f^*| - |f| > 0$, D has a member (P, P', δ), where P is a u_f-regular path from s to s'. Then P is an r-augmenting path for f. ■

In what follows we will use a simple construction which enables us to reduce the task of finding an r-augmenting path to the r-reachability problem. For a skew-symmetric network (H, h), split each arc $a = (x, y)$ of H into two parallel arcs a_1 and a_2 from x to y (the *first* and *second split-arcs* generated by a). These arcs are endowed with the capacities $[h](a_1) = \lceil h(a)/2 \rceil$ and $[h](a_2) = \lfloor h(a)/2 \rfloor$. Then delete all arcs with zero capacity $[h]$. The resulting capacitated graph is called the *split-graph* for (H, h) and denoted by $S(H, h)$. The symmetry σ is extended to the arcs of $S(H, h)$ in a natural way, by defining $\sigma(a_i) = (\sigma(a))_i$ for $i = 1, 2$.

For a path P in $S(H, h)$, its image in H is denoted by $\omega(P)$ (i.e., $\omega(P)$ is obtained by replacing each arc a_i of P by the original arc $a =: \omega(a_i)$). It is easy to see that if P is regular, then $\omega(P)$ is h-regular. Conversely, for any h-regular path Q in H, there is a (possibly not unique) r-path P in $S(H, h)$ such that $\omega(P) = Q$. Indeed, replace each ordinary arc a of Q by the first split-arc a_1 (existing as $h(a) \geq 1$) and replace each pair a, a' of arc mates in Q by a_i, a'_j for $\{i, j\} = \{1, 2\}$ (taking into account that $h(a) = h(a') \geq 2$). This gives the required r-path P. Thus, Theorem 3.3 admits the following re-formulation in terms of split-graphs.

Corollary 3.4 *An IS-flow f in (G, u) is maximum if and only if there is no regular path from s to s' in $S(G^+, u_f)$.*

Finally, the classic max-flow min-cut theorem states that the maximum flow value is equal to the minimum cut capacity. A skew-symmetric version of this theorem involves a more complicated object which is close to an s-barrier occurring in the solvability criterion for the r-reachability problem given in Theorem 2.1. We say that $\mathcal{B} = (A; X_1, \ldots, X_k)$ is an *odd s-barrier* for (G, u) if the following conditions hold.

(O1) $A, X_1, \ldots, X_k$ are pairwise disjoint subsets of V, and $s \in A$.

(O2) For $A' = \sigma(A)$, $A \cap A' = \emptyset$.

(O3) For $i = 1, \ldots, k$, X_i is symmetric, *i.e.*, $\sigma(X_i) = X_i$.

(O4) For $i = 1, \ldots, k$, $u(A, X_i)$ is odd.

(O5) For $i, j = 1, \ldots, k$ and $i \neq j$, no positive capacity arc connects X_i and X_j.

(O6) For $M = V - (A \cup A' \cup X_1 \cup \ldots \cup X_k)$ and $i = 1, \ldots, k$, no positive capacity arc connects X_i and M.

Compare with (B1)–(B7) in Section 2. We refer to an odd s-barrier $\mathcal{B} = (A; X_1, \ldots, X_k)$ as odd barrier, and define its *capacity* $u(\mathcal{B})$ to be $u(A, V - A) - k$.

The following is the symmetric max-flow min-cut theorem.

Theorem 3.5 *The maximum IS-flow value is equal to the minimum odd barrier capacity.*

Proof. To see that the capacity of an odd barrier $\mathcal{B} = (A; X_1, \ldots, X_k)$ is an upper bound on the value of an IS-flow f, consider a symmetric decomposition D of f. For each member (P, P', δ) of D, where P is a path from s to s', take the *last* arc $a = (x, y)$ of the *first* path P such that $x \in A$. If y is in some X_i, then (O1),(O2),(O5),(O6) imply that P leaves X_i by an arc b from X_i to A'. Then the symmetric arc b' belongs to the path P' and goes from A to X_i. Therefore, the elementary flow (P, P', δ) can be associated with the pair a, b' (possibly $a = b'$) and it uses at least 2δ units of the capacity $u(A, X_i)$. Since $u(A, X_i)$ is odd (by (O4)), at least one unit of this capacity is not used under the way we associate the elementary s–s' flows of D with arcs from A to $V - A$. This implies $|f| \leq u(\mathcal{B})$.

Next we show that the two values in the theorem are equal. Let f be a maximum IS-flow. By Corollary 3.4, the split-graph $S = S(G^+, u_f)$ contains no s–s' r-path, so it must contain an s-barrier $\mathcal{B} = (A; X_1, \ldots, X_k)$, by Theorem 2.1.

Let e^i be the (unique) arc from A to X_i in S (see (B4) in Section 2). By the construction of S, it follows that the residual capacity u_f of every arc from A to X_i in G^+ is zero except for the arc $\omega(e^i)$, whose residual capacity is one. Hence,

(i) if e^i was formed by splitting an arc $a \in E$, then a goes from A to X_i, and $f(a) = u(a) - 1$;

(ii) if e^i was formed by splitting a^R for $a \in E$, then a goes from X_i to A, and $f(a) = 1$;

(iii) all arcs from A to X_i in G, except a in case (i), are saturated by f;

(iv) all arcs from X_i to A in G, except a in case (ii), are free of flow.

Furthermore, comparing arcs in S and G, we observe that:

(v) property (B7) implies that the arcs from A to $A' \cup M$ are saturated and the arcs from $A' \cup M$ to A are free of flow;

(vi) property (B5) implies (O5) and (B6) implies (O6).

¿From (i)-(vi) it follows that $\mathcal{B}$ is an odd s-barrier in G and $|f| = u(\mathcal{B})$. ∎

4 Integer and Linear Programming Formulations

We can state the MSFP as an integer program in a straightforward way. We use function rather than vector notation. Given two functions g and h on a set S, $g \cdot h$ denotes $\sum_{x \in S} g(x)h(x)$.

The integer program corresponding to the MSFP is as follows, assuming that no arc of G enters s (by Remark in the previous section).

$$\textbf{maximize} \quad |f| = \sum_{(s,v) \in E} f(s,v) \tag{4.1}$$

$$\textbf{subject to}$$

$$f(a) \geq 0 \qquad \forall a \in E \tag{4.2}$$

$$f(a) \leq u(a) \qquad \forall a \in E \tag{4.3}$$

$$\sum_{(u,v) \in E} f(u,v) - \sum_{(v,w) \in E} f(v,w) = 0 \qquad \forall v \in V - \{s,t\} \tag{4.4}$$

$$f(a) - f(\sigma(a)) = 0 \qquad \forall a \in E \tag{4.5}$$

$$f(a) \ \text{ integral} \qquad \forall a \in E \tag{4.6}$$

We obtain an alternative linear programming formulation for the MSFP by replacing the integrality condition (4.6) by certain linear constraints related to so-called fragments. This linear program and its dual (discussed below) are analogous to, but somewhat more complicated than, those for the usual maximum flow problem and its dual in [9]; although we will not use explicitly these programs in our methods, they deserve to be discussed in their own right.

An *odd fragment* is a pair $\rho = (V_\rho, U_\rho)$, where V_ρ is a symmetric set of nodes with $s \notin V_\rho$, and U_ρ is a subset of arcs entering V_ρ such that the total capacity $u(U_\rho)$ is odd. The *characteristic function* χ_ρ of ρ is the function on E defined by

$$\chi_\rho(a) = \begin{cases} 1 & \text{if } a \in U_\rho \cup \sigma(U_\rho), \\ -1 & \text{if } a \in \delta(V_\rho) - (U_\rho \cup \sigma(U_\rho)), \\ 0 & \text{otherwise.} \end{cases} \tag{4.7}$$

Here $\delta(V_\rho)$ is the set of arcs with one end in V_ρ and the other in $V - V_\rho$. We denote the set of odd fragments by Ω.

Let f be a (feasible) IS-flow and $\rho \in \Omega$. Since u is symmetric, the definition of χ_ρ shows that $f \cdot \chi_\rho \leq 2u(U_\rho)$. Moreover, $f \cdot \chi_\rho$ is at most $2u(U_\rho) - 2$, as it follows immediately from Corollary 3.2 and the fact that $u(U_\rho)$ is odd. This gives additional linear constraints for the MSFP:

$$f \cdot \chi_\rho \leq 2u(U_\rho) - 2 \quad \text{for each } \ \rho \in \Omega. \tag{4.8}$$

It turns out that adding these constraints, we can drop off the symmetry constraints (4.5) and the integrality constraints (4.6) without changing the optimum value of the linear program. This fact is implied by the following theorem.

Theorem 4.1 *Every maximum IS-flow is an optimal solution to the linear program* (4.1)–(4.4), (4.8).

Proof. Assign a dual variable $\pi(v) \in \mathbf{R}$ (a *potential*) to each node $v \in V$, $\gamma(a) \in \mathbf{R}_+$ (a *length*) to each arc $a \in E$, and $\xi(\rho) \in \mathbf{R}_+$ to each odd fragment $\rho \in \Omega$. For an arc $a = (v, w)$, define $\Delta_\pi(a) = \pi(w) - \pi(v)$ (the *potential difference*), and $q_{\gamma,\xi}(a) = \gamma(a) + \sum_\Omega \xi(\rho)\chi_\rho(a)$.

Consider the linear program:

$$\begin{aligned} &\textbf{minimize} \quad \psi(\pi,\gamma,\xi) && = \sum_E u(a)\gamma(a) + \sum_\Omega (2u(U_\rho) - 2)\xi(\rho) && (4.9)\\ &\textbf{subject to} \\ &\gamma(a) \geq 0 && \forall a \in E && (4.10)\\ &\xi(\rho) \geq 0 && \forall \rho \in \Omega && (4.11)\\ &\pi(s) = 0 && && (4.12)\\ &\pi(s') = 1 && && (4.13)\\ &q_{\gamma,\xi}(a) - \Delta_\pi(a) \geq 0 && \forall a \in E. && (4.14) \end{aligned}$$

One can see that (4.1)–(4.4),(4.8) and (4.9)–(4.14) are mutually dual linear programs (if we formally introduce an extra arc (s', s), add the conservation constraints for s and s', and replace the objective (4.1) by $\max\{f(s', s)\}$). Therefore,

$$\max |f| = \min \psi(\pi, \gamma, \xi), \tag{4.15}$$

where the maximum and minimum range over the feasible solutions to these programs, respectively.

We assert that every maximum IS-flow f achieves the maximum in (4.15). To see this, choose an odd barrier $\mathcal{B} = (A; X_1, \ldots, X_k)$ of minimum capacity $u(\mathcal{B})$. For $i = 1, \ldots, k$, let U_i be the set of arcs from A to X_i; then $\rho_i = (X_i, U_i)$ is an odd fragment for G, u. Define $\pi(v)$ to be 0 for $v \in A$, 1 for $v \in A'$, and 1/2 otherwise. Define $\gamma(a)$ to be 1 for $a \in (A, A')$, 1/2 for $a \in (A, M) \cup (M, A')$, and 0 otherwise, where $M = V - (A \cup A' \cup X_1 \cup \ldots \cup X_k)$. Define $\xi(\rho_i) = 1/2$ for $i = 1, \ldots, k$, and $\xi(\rho) = 0$ for the other odd fragments in (G, u).

One can check that (4.14) holds for all a (*e.g.*, $q_{\gamma,\xi}(a) = \Delta_\pi(a) = 1$ for $a \in (A, A')$ and $q_{\gamma,\xi}(a) = \Delta_\pi(a) = 1/2$ for $a \in (A, L) \cup (L, A')$, where $L = V - (A \cup A')$). Thus π, γ, ξ are feasible.

Using the fact that $u(A, M) = u(M, A')$, we observe that $u \cdot \gamma = u(A, A') + u(A, M)$. Also

$$\sum_\Omega (2u(U_\rho) - 2)\xi(\rho) = \sum_{i=1}^{k} \frac{1}{2}(2u(U_i) - 2) = \left(\sum_{i=1}^{k} u(A, X_k)\right) - k.$$

This implies $\psi(\pi, \gamma, \xi) = u(\mathcal{B})$, and now the result follows from Theorem 3.5. ∎

Given potentials $\pi(v)$, $v \in V$, and a function $\xi : \Omega \to \mathbf{R}_+$, define the *cost* $c_{\pi,\xi}(a)$ of each arc $a = (v, w) \in E$ by

$$c_{\pi,\xi}(a) = \pi(w) - \pi(v) - \sum_\Omega \xi(\rho)\chi_\rho(a).$$

Using such costs, we can get rid of the dual variables γ in (4.9)–(4.14) by doing the substitution $\gamma(a) = \max\{0, c_{\pi,\xi}(a)\}$. This gives an optimality criterion for the MSFP as follows.

Theorem 4.2 *An IS-flow f is maximum if and only if there are a potential function $\pi : V \to \mathbf{R}$ with $\pi(s') - \pi(s) = 1$ and a function $\xi : \Omega \to \mathbf{R}_+$ such that the following "complementary slackness" conditions hold:*

(CS1) for $\rho \in \Omega$, $\xi(\rho) > 0$ implies $f \cdot \chi_\rho = 2u(U_\rho) - 2$;

(CS2) for $a \in E$, $c_{\pi,\xi}(a) > 0$ implies $f(a) = u(a)$;

(CS3) for $a \in E$, $c_{\pi,\xi}(a) < 0$ implies $f(a) = 0$.

5 Algorithm Using a Good Initial Solution

In this section we describe a relatively simple algorithm to solve the maximum IS-flow problem in a skew-symmetric network $N = (G, u)$. It finds a "nearly optimal" IS-flow and then makes only $O(n)$ augmentations to obtain a maximum IS-flow. The algorithm consists of four stages.

The *first* stage ignores the fact that N is skew-symmetric and finds an integer maximum flow g in N by use of one or another max-flow algorithm. Then we put $h(a) = (g(a) + g(a'))/2$ for all arcs $a \in E$. Since $\text{div}_h(s) = \text{div}_g(s)/2 - \text{div}_g(s')/2 = \text{div}_g(s)$, h is a maximum flow as well. Also h is symmetric and its values on the arcs are multiple of $1/2$. Let Z be the set of arcs on which h is not integral. If $Z = \emptyset$, then h is already a maximum IS-flow; so assume this is not the case.

The *second* stage attempts to improve h by decreasing Z. Let $H = (X, Z)$ be the subgraph of G induced by Z. For each $x \in V$, the half-integrality of h and the equality $\text{div}_g(x) + \text{div}_g(x') = 0$ imply that x is incident to an even number of arcs in Z. Therefore, we can decompose H into simple, not necessarily directed, cycles $C_1, \ldots, C_r$ which are pairwise arc-disjoint. Moreover, we can find, in linear time, a decomposition in which each cycle C_i is either self-symmetric ($C_i = \sigma(C_i)$) or symmetric to another cycle C_j ($C_i = \sigma(C_j)$).

To do this, we start with some node $v_0 \in X$ and grow in H a simple (undirected) path $P = (v_0, e_1, v_1, \ldots, e_q, v_q)$ such that the mate v_i' of each node v_i is not in P. At each step, we choose in H an arc $e \neq e_q$ incident to the last node v_q (obviously, such an arc exists); let x be the other end node of e. If none of x, x' is in P, then we add e to P. If some of x, x' is a node of P, v_i say, then we shorten P by removing its end part from e_{i+1} and delete from H the arcs $e_{i+1}, \ldots, e_q, e$ and their mates. One can see that the arcs deleted induce a self-symmetric cycle (when $x' = v_i$) or two disjoint symmetric cycles (when $x = v_i$). We also remove the isolated nodes created by the arc deletions and change the initial node v_0 if needed. Repeating the process for the new current graph H and path P, we eventually obtain the desired decomposition $\mathcal{C}$, in $O(|Z|)$ time.

Next we examine the cycles in $\mathcal{C}$. Each pair C, C' of symmetric cycles is canceled by sending a half unit of flow through C and through C', i.e., we increase (resp. decrease) $h(e)$ by $1/2$ on each forward (resp. backward) arc e of these cycles. The resulting function h is symmetric, and $\text{div}_h(x)$

preserves at each node x, whence h is again a maximum symmetric flow. Now suppose that two self-symmetric cycles C and D meet at a node x. Then they meet at x' as well. Concatenating the x–x' path in C and the x'–x path in D and concatenating the rests of C and D, we obtain a pair of symmetric cycles and cancel these cycles as above.

These cancellations result in $\mathcal{C}$ consisting of pairwise *node-disjoint* self-symmetric cycles, say $C_1, \dots, C_k$. The second stage takes $O(m)$ time.

The *third* stage transforms h into an IS-flow f whose value $|f|$ is at most k units below $|h|$. For each i, fix a node t_i in C_i and change h on C_i by sending a half unit of flow through the t_i–t'_i path in C_i and through the reverse to the t'_i–t_i path in it. The resulting function h is integer and symmetric and the divergences preserve at all nodes except for the nodes t_i and t'_i where we have $\text{div}_h(t_i) = -\text{div}_h(t'_i) = 1$ for each i (assuming w.l.o.g. that all t_i's are different from s'). Therefore, h is, in essence, a multiterminal IS-flow with sources $s, t_1, \dots, t_k$ and sinks $s', t'_1, \dots, t'_k$. A genuine IS-flow f from s to s' is extracted by reducing h on some h-regular paths. More precisely, we add to G artificial arcs $e_i = (s, t_i)$, $i = 1, \dots, k$ and their mates, extend h by ones to these arcs and construct a symmetric decomposition $\mathcal{D}$ (defined in Section 3) for the obtained function h' in the resulting graph G' (clearly h' is an IS-flow of value $|h| + k$).

Let $\mathcal{D}'$ be the set of elementary flows in $\mathcal{D}$ formed by the paths or cycles which contain artificial arcs. Then $\delta = 1$ for each $(P, P', \delta) \in \mathcal{D}'$. Define $f' = h' - \sum(\chi^P + \chi^{P'} : (P, P', 1) \in \mathcal{D}')$. Then f' is an IS-flow in G', and $|f'| \geq |h'| - 2k \geq |h| - k$. Moreover, since $f(e_i) = 0$ for $i = 1, \dots, k$, the restriction f of f' to E is an IS-flow in G, and $|f| = |f'|$. Thus, $|f| \geq |h| - k$, and now the facts that $k \leq n/2$ (as the nodes $t_1, \dots, t_k, t'_1, \dots, t'_k$ are different) and that h is a maximum flow in N imply that the value of f differs from the maximum IS-flow value by $O(n)$. The third stage takes $O(nm)$ time (the time needed to construct a symmetric decomposition of h').

The final, *fourth*, stage transforms f into a maximum IS-flow. Each iteration applies the r-reachability algorithm (RA) mentioned in Section 2 to the split-graph $S(G^+, u_f)$ in order to find a u_f-regular s–s' path P in G^+ and then augment the current IS-flow f by the elementary flow $(P, P', \delta_{u_f}(P))$ as explained in Section 3. Thus, a maximum IS-flow in N is constructed in $O(n)$ iterations. Since the RA runs in $O(m)$ time (by Theorem 2.2), the fourth stage takes $O(nm)$ time.

Summing up the above arguments, we conclude with the following.

Theorem 5.1 *The above algorithm finds a maximum IS-flow in N in $O(M(n,m) + nm)$ time, where $M(n,m)$ is the running time of the max-flow procedure it applies.*

6 Shortest R-Augmenting Paths and Blocking IS-Flows

Theorem 3.3 and Corollary 3.4 prompt an alternative method for finding a maximum IS-flow in a skew-symmetric network $N = (G, u)$, which is analogous to the method of Ford and Fulkerson for usual flows. It starts with the zero flow, and at each iteration, the current IS-flow f is augmented by an elementary flow in (G^+, u_f) (found by applying the r-reachability algorithm to $S(G^+, u_f)$).

Since each iteration increases the value of f by at least two, a maximum IS-flow is constructed in pseudo-polynomial time. In general, this method is not competitive to the method of Section 5.

More efficient methods involve the concepts of shortest r-augmenting paths and shortest blocking IS-flows that we now introduce. Let g be an IS-flow in a skew-symmetric network $(H = (V, W), h)$. Let $g(W)$ stand for $\sum_{e\in W} g(e)$ (the *volume* of g). Considering a symmetric decomposition $D = \{(P_i, P_i', \delta_i) : i = 1, \dots, k)$ of g, we have

$$g(W) = \sum(\delta_i|P_i| + \delta_i|P_i'| : i = 1, \dots, k) \geq |g| \min\{|P_i| : i = 1, \dots, k\}.$$

This implies

$$g(W) \geq |g| \text{r-dist}_{S(H,h)}(s, s'), \tag{6.1}$$

where $\text{r-dist}_{H'}(x, y)$ denotes the minimum length of a regular x–y path in a skew-symmetric graph H' (the *regular distance* from x to y). We say that an IS-flow g is

(i) *shortest* if (6.1) holds with equality, i.e., some (equivalently, any) symmetric decomposition of g consists of shortest h-regular paths from s to s';

(ii) *totally blocking* if there is no $(h-g)$-regular path from s to s' in H, i.e., we cannot augment g using only residual capacities in H itself;

(iii) *shortest blocking* if g is shortest (as in (i)) and

$$\text{r-dist}_{S(H,h-g)}(s, s') > \text{r-dist}_{S(H,h)}(s, s'). \tag{6.2}$$

Note that a shortest blocking IS-flow is not necessarily totally blocking, and vice versa.

Given a skew-symmetric network $N = (G, u)$, the *shortest r-augmenting path method (SAPM)*, analogous to the method of Edmonds and Karp [7] for usual flows, starts with the zero flow, and each iteration augments the current IS-flow f by a shortest elementary flow $g = (P, P', \delta_{u_f}(P))$.

The *shortest blocking IS-flow method (SBFM)*, analogous to Dinitz' method [4], starts with the zero flow, and each *big iteration* augments the current IS-flow f by performing the following two steps.

(P1) Find a shortest blocking IS-flow g in (G^+, u_f).

(P2) Update $f := f \oplus g$.

Both methods terminate when f no longer admits r-augmenting paths (i.e., g becomes the zero flow). The following observation is crucial for the complexities of the methods.

Lemma 6.1 *Let g be a shortest IS-flow in (G^+, u_f), and let $f' = f \oplus g$. Let k and k' be the minimum lengths of r-augmenting paths for f and f', respectively. Then $k' \geq k$. Moreover, if g is a shortest blocking IS-flow, then $k' > k$.*

Proof. Take a shortest $u_{f'}$-regular path P from s to s' in G^+. Then $|P| = k'$ and $g' = (P, P', 1)$ is an elementary flow in $(G^+, u_{f'})$.

Note that supp(g) does not contain opposed arcs $a = (x, y)$ and $b = (y, x)$. Otherwise decreasing g by one on each of a, b, a', b' (which are, obviously, different), we would obtain the IS-flow $\tilde{g}$ in (G^+, u_f) such that $|\tilde{g}| = |g|$ and $\tilde{g}(E^+) < g(E^+)$, which is impossible because $\tilde{g}(E^+) \geq k|\tilde{g}|$ and $g(E^+) = k|g|$. This implies that each arc a in the set $Z = \{a \in E^+ : g(a^R) = 0\}$ satisfies

$$u_{f'}(a) = u_f(a) - g(a). \tag{6.3}$$

If supp(g') $\subseteq Z$, then g' is a feasible IS-flow in (G^+, u_f) (by (6.3)), whence $k' = g'(E^+)/|g'| \geq k$. Moreover, if, in addition, g is a shortest blocking IS-flow, then (6.2) and the fact that $g' \leq u_f - g$ (by (6.3)) imply $k' > k$.

Now suppose there is an arc $e \in E^+$ such that $g'(e) > 0$ and $g(e^R) > 0$. For each $a \in E^+$, put $\lambda(a) = \max\{0, g(a) + g'(a) - g(a^R) - g'(a^R)\}$. One can check that $\lambda(a) \leq u_f(a)$ for all arcs a. Therefore, λ is an IS-flow in (G^+, u_f) with $|\lambda| = |g| + |g'| = |g| + 2$. Also $\lambda(E^+) < g(E^+) + g'(E^+)$ since for the e above, $\lambda(e) + \lambda(e^R) < g'(e) + g(e^R)$. We have

$$2k' = g'(E^+) > \lambda(E^+) - g(E^+) \geq k(|g| + 2) - k|g| = 2k,$$

yielding $k' > k$. ∎

Thus, each iteration of the SAPM does not decrease the minimum length of an r-augmenting path, while each big iteration of the SBFM increases such a length. This gives upper bounds on the numbers of iterations.

Corollary 6.2 *SAPM terminates in at most* $(n-1)m$ *iterations.*

(This uses the fact (seen from the proof of Lemma 6.1) that in the sequence of iterations with the same length of shortest r-augmenting paths, the subgraph of G^+ induced by the arcs contained in such paths is monotone non-increasing, and each iteration reduces the capacity of some arc of this subgraph, as well as the capacity of its mate, to zero or one.)

Corollary 6.3 *SBFM terminates in at most* $n-1$ *big iterations.*

As mentioned above, the SBFM can be considered as a skew-symmetric analog of Dinitz' blocking flow algorithm. Recall that each (big) iteration of that algorithm constructs a blocking flow in the subnetwork H formed by the nodes and arcs of shortest augmenting paths. Such a network is acyclic (moreover, layered), and a blocking flow in H is easily constructed in $O(nm)$ time.

The problem of finding a shortest blocking IS-flow ((P1) above) is more complicated. Let H be the subgraph of G^+ formed by the nodes and arcs contained in shortest u_f-regular s–s' paths. Such an H needs not be acyclic (counterexamples are not difficult). We will show that problem (P1) can be reduced to a seemingly easier task, namely, to finding a totally blocking IS-flow in a certain acyclic network $(\overline{H}, \overline{h})$. Such a network arises when the shortest r-path algorithm from [14] is applied to the split-graph $S(G^+, u_f)$ with unit lengths of the arcs.

In order to show this, in Section 8, we first need to review the r-reachability and shortest r-path algorithms.

7 Regular and Shortest Regular Paths

In this section we review the r-reachability and shortest r-path algorithms, referring the reader to [14] for details. The former algorithm is based on a so-called bud trimming operation, which is analogous to cutting blossoms in matching algorithms.

7.1 Buds and Trimming Operation

A *bud* is a triple $\tau = (V_\tau, E_\tau, e_\tau = (v, w))$ such that

(1) (V_τ, E_τ) is a symmetric subgraph of G with $s \notin V_\tau$.

(2) e_τ is an arc of G entering V_τ, *i.e.*, $v \notin V_\tau \ni w$.

(3) For each node $x \in V_\tau$, there is an r-path from w to x in (V_τ, E_τ) (and therefore an r-path from x to $w' = \sigma(w)$).

(4) There is an r-path from s to v which meets V_τ only at v.

The node w is called the *base node* of τ and the arc e_τ is called the *base arc*. The node w' is called the *anti-base node* and the arc e'_τ is called the *anti-base arc*. A bud τ is called *elementary* if (V_τ, E_τ) is the union of an r-path from w to w' and its symmetric path (also from w to w').

Given a bud τ with $e_\tau = (v, w)$, the *trimming operation* transforms G into $\overline{G}$ with the node set

$$\overline{V} = V - (V_\tau - \{w, w'\})$$

and arc set $\overline{E}$ constructed as follows.

1. Each arc $a = (x, y) \in E$ such that either $x, y \in V - V_\tau$ or $a = e_\tau$ or $a = e'_\tau$ remains in $\overline{E}$.
2. Each arc $(x, y) \in E - \{e'_\tau\}$ that leaves V_τ is replaced by an arc from w to y.
3. Each arc $(x, y) \in E - \{e_\tau\}$ that enters V_τ is replaced by an arc from x to w'.
4. Each arc $e \notin E_\tau$ with both ends in V_τ is replaced by an arc from w to w'.

See Figure 2 for an example of bud trimming. Clearly $\overline{G}$ has a naturally induced skew-symmetry. We identify each arc in $\overline{E}$ with the corresponding arc in E (the ends of an arc can change).

An important property is that the bud trimming operation preserves the regular reachability from s to s'.

Lemma 7.1 [14] *There is an r-path from s to s' in G if and only if there is an r-path from s to s' in $\overline{G}$.*

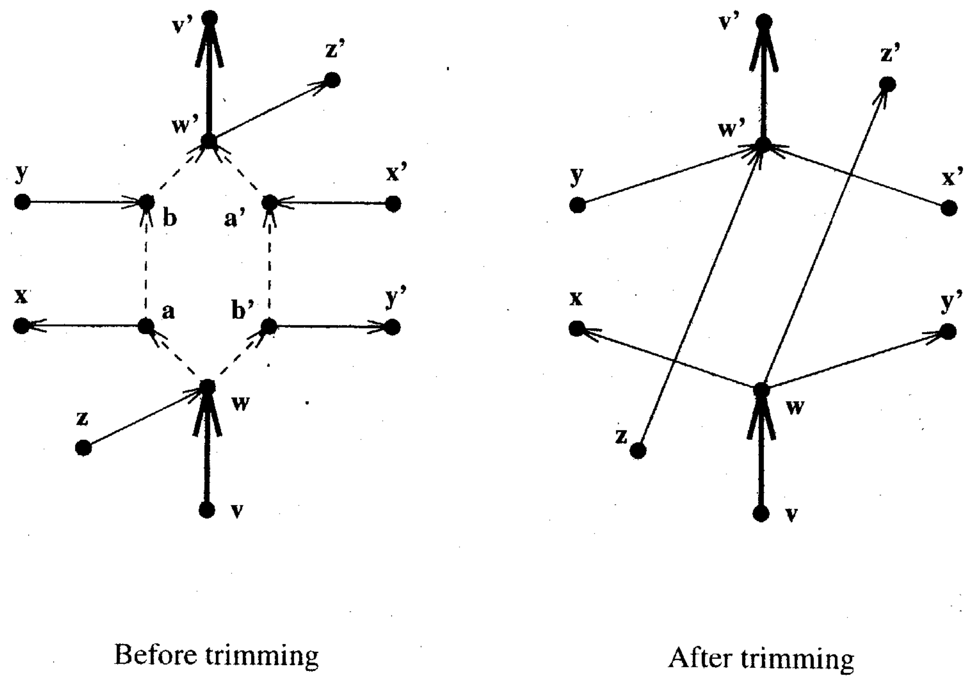


Figure 2: *Bud trimming example.*

7.2 The Regular Reachability Algorithm (RA)

The algorithm searches for a regular s–s' path in a given skew-symmetric graph $\Gamma = (V, W)$. It grows a *tree* (A, T) *rooted at* s such that every rooted path in this tree is regular, and the sets A and $A' = \sigma(A)$ are disjoint. By symmetry, $(A', T' = \sigma(T))$ is a tree rooted to s'. Initially $A = \{s\}$ and $T = \emptyset$.

Each iteration chooses an arc $e = (x, y)$ with $x \in A$ and $y \notin A$ for the current graph Γ and tree (A, T). If $y \notin A'$, then y is added to A, e to T, y' to A', and e' to T'. If $y \in A'$, then the algorithm checks the concatenation P of the s–x path P_1 in (A, T), the arc e, and the y–s' path P_2 in (A', T'). If P is regular, the algorithm terminates.

Suppose P is not regular. Let (v, w) be the arc of P_1 closest to x such that the symmetric arc (w', v') is on P_2. Let Q be the subpath of P between w and w'. Then $Q \cup Q'$ forms an elementary bud τ in Γ with the base arc (v, w). The algorithm trims τ (producing the new current graph Γ) and accordingly updates the tree (A, T) (by shrinking the corresponding arcs in $Q \cup Q'$). Finally, if there is no arc e as above, then the algorithm terminates with the conclusion that no regular s–s' path exists in the input graph Γ_0; this follows from Lemma 7.1. Moreover, considering the resulting set A and the maximal trimmed buds of the final graph Γ, one can construct an s-barrier in Γ_0.

Note that the r-path P found by the algorithm is a path in the current graph, possibly repeatedly trimmed many times. The postprocessing stage of the algorithm transforms P into an r-path from s to s' in the initial graph Γ_0. This is performed by a *restoration procedure*, as follows. If P is already a path of Γ_0, we are done. Otherwise P contains two consecutive arcs a, b such that either a is the base arc or b is the anti-base arc of some maximal trimmed bud τ. Then we "undo" the corresponding trimming operation applied, connecting the head of a to the tail of b by an r-path in (V_τ, E_τ). This transforms P into a regular r-path in the new ("undone") current graph Γ, and we continue the process until a regular s–s' path in Γ_0 is obtained. Being supported by certain data structures created by the RA, the restoration procedure can be implemented in time linear in $|E_{\tau(1)}| + \ldots + |E_{\tau(k)}|$, where $\tau(1), \ldots, \tau(k)$ are the buds whose base or anti-base arcs occur in the current paths P. This gives $O(|W|)$ time for the restoration procedure since the arc sets of the trimmed buds are disjoint. Moreover, one shows that the running time is, in fact, $O(|V|)$.

A fast implementation of the RA finds a regular s–s' path or an s-barrier in linear time, as indicated in Theorem 2.2.

7.3 The Shortest Regular Path Algorithm (SPA)

We now consider the shortest regular path problem (SPP) in a skew-symmetric graph $\Gamma = (V, W)$ with *nonnegative* symmetric lengths $\ell(e)$ of the arcs $e \in W$. Its dual problem involves so-called fragments, which are close to, but somewhat different from, odd fragments introduced in Section 4. A *fragment* is a pair $\phi = (V_\phi, e_\phi = (v, w))$, where V_ϕ is a symmetric set of nodes of Γ with $s \notin V_\phi$ and e_ϕ is an arc entering V_ϕ. For instance, every bud induces the fragment defined by its node set and its base arc. We call w, w', e_ϕ, e'_ϕ the *base node*, *anti-base node*, *base arc* and *anti-base arc* of ϕ, respectively, and define the characteristic function χ_ϕ of ϕ by

$$\chi_\phi(a) = \begin{cases} 1 & \text{for } a = e_\phi, e'_\phi, \\ -1 & \text{for } a \in \delta(V_\phi) - \{e_\phi, e'_\phi\}, \\ 0 & \text{for the remaining arcs of } \Gamma. \end{cases} \tag{7.1}$$

Compare with (4.7). One can see that $\chi^P \cdot \chi_\phi \leq 0$ holds for any r-cycle P and any r-path P connecting symmetric nodes, and this turns into equality if and only if $P \cap \delta(V_\phi)$ either is empty, or consists of two arcs one of which is e_ϕ or e'_ϕ.

For a function $\pi : V \to \mathbf{R}$ (of node *potentials*) and a nonnegative function ξ on a set Φ of fragments, define the *modified length* of an arc $e = (x, y)$ to be

$$\ell^\xi_\pi(e) = \ell(e) + \pi(x) - \pi(y) + \sum_{\phi \in \Phi} \xi(\phi)\chi_\phi(e).$$

The duality theorem for SPP can be formulated as follows.

Theorem 7.2 [14] *A regular path P from s to s' is a shortest r-path if and only if there exist a potential $\pi : V \to \mathbf{R}$ and a* positive *function ξ on a set Φ of fragments such that*

$$\ell^\xi_\pi(e) \geq 0 \quad \textit{for each} \quad e \in W; \tag{7.2}$$

$$\ell^\xi_\pi(e) = 0 \quad \textit{for each} \quad e \in P; \tag{7.3}$$

$$\chi^P \cdot \chi_\phi = 0 \quad \textit{for each} \quad \phi \in \Phi. \tag{7.4}$$

The *shortest r-path algorithm (SPA)* from [14] implicitly maintains π, Φ, ξ and iteratively transforms the input graph Γ, starting with $\Gamma_1 = \Gamma$, $\pi = 0$ and $\Phi = \emptyset$. Each, ith, iteration deals with a graph Γ_i; for $i > 1$, Γ_i is obtained by trimming some buds in the previous graph Γ_{i-1}. Let Γ_i^0 denote the subgraph of Γ_i induced by the arcs with zero modified length; we call Γ_i^0 the current *0-graph* (recall that the arcs of any trimmed graph are identified with the corresponding arcs of the original graph). The iteration applies the above r-reachability algorithm to search for a regular s–s' path in Γ_i^0 (whose arc lists are efficiently accessible, in spite of the fact that the modified lengths of arcs in Γ_i are maintained implicitly). If such a path is found, the algorithm terminates and outputs this path to a postprocessing stage described below.

If such a path does not exist, the RA constructs an s-barrier $(A; X_1, \ldots, X_k)$ in Γ_i^0. Moreover, each X_j is the node set of a bud τ whose base arc e_τ is just the (unique) arc going from A to X_j. Such a τ generates (implicitly) the fragment ϕ with $e_\phi = e_\tau$ and V_ϕ to be the set of preimages of elements of X_j in Γ. The iteration adds these fragments ϕ to the current set Φ, assigns certain uniform weights $\xi(\phi) = \epsilon > 0$ to these ϕ's, increases by ϵ the weights of all maximal fragments in Φ whose base nodes are in A, and updates the potential π by $\pi(x) := \pi(x) - \epsilon$ and $\pi(x') = \pi(x') + \epsilon$ for all $x \in V$ such that x belongs to A and differs from the base nodes of maximal fragments in Φ. Then the iteration trims the above buds τ, giving the input graph Γ_{i+1} for the next iteration. The process terminates when some, qth, iteration finds a regular s–s' path $\overline{P}$ in the current 0-graph Γ_q^0.

The postprocessing stage applies the restoration procedure of the RA, mentioned in Subsection 7.2, to extend $\overline{P}$ into a regular s–s' path P in the initial graph Γ. All arcs of this P have zero modified lengths (since the arcs of all buds are such). Moreover, the restoration procedure ensures that $\chi^P \cdot \chi_\phi = 0$ for all $\phi \in \Phi$. Thus, P is the desired shortest s–s' path for Γ, ℓ, by Theorem 7.2.

In fact, the dual objects π, Φ, ξ constructed this way have certain sharper properties. These properties, exhibited in the theorem below, will be essentially used in the next section. For the π, Φ, ξ obtained, define $W^0 = \{e \in W : \ell_\pi^\xi(e) = 0\}$ and call the subgraph $\Gamma^0 = (V, W^0)$ of Γ the *full 0-graph*. The final graph Γ_q^0 is denoted by $\overline{\Gamma}^0 = (\overline{V}, \overline{W}^0)$ and called the *trimmed 0-graph*; in fact, this graph is explicitly constructed by the algorithm. A path Q in Γ^0 is called *well-crossing* (the fragments in Φ) if Q is an r-path satisfying $\chi^Q \cdot \chi_\phi \geq 0$ for each $\phi \in \Phi$ (i.e., every nonempty set $Q \cap \delta(V_\phi)$ consists of at most two arcs one of which is e_ϕ or e'_ϕ). By Theorem 7.2, the well-crossing s–s' paths are exactly the shortest regular s–s' paths for Γ, ℓ.

Theorem 7.3 [14] *SPA runs in $O(m \log n)$ time, and in $O(n\sqrt{\log L})$ time when ℓ is integer-valued and L is the maximum arc length. Furthermore, the algorithm constructs (implicitly) π, Φ, ξ as in Theorem 7.2 which possess the following additional properties:*

(A1) π *is* anti-symmetric, *i.e.,* $\pi(x) = -\pi(x')$ *for all* $x \in V$*;*

(A2) Φ *is* nested, *in the sense that any distinct* $\phi, \phi' \in \Phi$ *satisfy either* $V_\phi \subset V_{\phi'}$ *or* $V_{\phi'} \subset V_\phi$ *or* $V_\phi \cap V_{\phi'} = \emptyset$*;*

(A3) *for* $\phi, \phi' \in \Phi$, *if* $V_{\phi'} \subset V_\phi$ *and* $e_{\phi'} \in \delta(V_\phi)$, *then* $e_{\phi'} = e_\phi$;

(A4) *for* $\phi, \phi' \in \Phi$, *if* $V_\phi \cap V_{\phi'} = \emptyset$ *and* $e_{\phi'} \in \delta(V_\phi)$, *then* $e_\phi \neq \delta(V_{\phi'})$;

(A5) *for each* $\phi \in \Phi$, $\ell^\xi_\pi(e_\phi) = 0$ *and the tail of* e_ϕ *is reachable from* s *by a well-crossing path* $Q(x)$ *in* Γ^0 *disjoint from* V_ϕ;

(A6) *for each* $\phi \in \Phi$, *each node* $x \in V_\phi$ *is reachable from the head of* e_ϕ *by a well-crossing path* $Q_\phi(x)$ *in* Γ^0 *with all nodes in* V_ϕ; *such a path can be found by the restoration procedure of the RA in* $O(|V_\phi|)$ *time.*

Note that the modified length function ℓ^ξ_π is symmetric. This is because each arc $e = (x, y)$ satisfies $\ell(e) = \ell(e')$, $\chi_\phi(e) = \chi_\phi(e')$ for any $\phi \in \Phi$ (by (7.1)) and $\pi(x) - \pi(y) = \pi(y') - \pi(x')$ (since ϕ is anti-symmetric by (A1)). Therefore, the graphs Γ^0 and $\overline{\Gamma}^0$ are skew-symmetric.

In view of (A5)–(A6), for each $\phi \in \Phi$, the triple $(V_\phi, W^0_\phi, e_\phi)$ forms a bud in Γ^0, denoted by $\tau(\phi)$, where W^0_ϕ is the set of arcs of Γ^0 with both ends in V_ϕ (we emphasize that $\tau(\phi)$ is a bud in the initial graph, while the bud generating ϕ in the algorithm concerns a current, possibly trimmed, graph). Let $\Phi^{\max}$ be the set of maximal fragments in the partial order on Φ defined by setting $\phi \succ \phi'$ for $V_\phi \supset V_{\phi'}$. Since Φ is nested (by (A2)), the sets V_ϕ for $\phi \in \Phi^{\max}$ are pairwise disjoint. One can see that the graph obtained from Γ^0 by simultaneously trimming the buds $\tau(\phi)$ for all $\phi \in \Phi^{\max}$ is precisely the trimmed 0-graph $\overline{\Gamma}^0$.

In the next section we will take advantage of a relationship between r-paths in $\overline{\Gamma}^0$ and shortest r-paths in (Γ, ℓ). More precisely, any well-crossing s–s' path in Γ^0 becomes a regular s–s' path in $\overline{\Gamma}^0$ under trimming the buds $\tau(\phi)$ for all $\phi \in \Phi^{\max}$. Conversely, let $\overline{Q}$ be a regular s–s' path in $\overline{\Gamma}^0$. If e, a are consecutive arcs in $\overline{Q}$ which share the base node w of a trimmed bud $\tau(\phi)$, then $e = e_\phi$; therefore, we can connect in Γ^0 the node w with the tail x of a by a path $Q_\phi(x)$ as in (A6). The fact that $Q_\phi(x)$ is well-crossing implies that the concatenation R of e, $Q_\phi(x)$ and a obeys $\chi^R \cdot \chi_{\phi'} = 0$ for each $\phi' \preceq \phi$ (taking into account (A3)). Similarly, if e, a share the anti-base node of $\tau(\phi)$, we use the corresponding symmetric path $Q'_\phi(x)$ in Γ^0_ϕ (which exists as W^0_ϕ is symmetric). Doing so for every such pair e, a in $\overline{Q}$, we eventually obtain an s–s' r-path Q in Γ^0. Then $\chi^Q \cdot \chi_\phi = 0$ for each $\phi \in \Phi$, whence Q is a shortest s–s' r-path.

Another important property of $\overline{\Gamma}^0$ is as follows.

Lemma 7.4 *If the length* $\ell(C)$ *of every r-cycle* C *in* Γ *is positive, then* $\overline{\Gamma}^0$ *is acyclic. In particular,* $\overline{\Gamma}^0$ *is acyclic if* $\ell(e) > 0$ *for all* $e \in E$.

Proof. Suppose this is not so, and let $\overline{C}$ be a simple cycle in $\overline{\Gamma}^0$. By the argument above, we can extend $\overline{C}$ into a well-crossing cycle C of Γ^0 (by adding corresponding well-crossing paths in the subgraphs of Γ^0 induced by the maximal fragments ϕ with $|\delta(V_\phi) \cap \overline{C}| \neq \emptyset$). Then $\chi^C \cdot \chi_\phi = 0$ for each $\phi \in \Phi$. This implies that the original length $\ell(C)$ and the modified length $\ell^\xi_\pi(C)$ are the same, since the changes in ℓ^ξ_π due to π cancel out as we go around the cycle. Hence, $\ell(C) = \ell^\xi_\pi(C) = 0$, contrary to the hypotheses of the lemma. ■

8 Reduction to an Acyclic Network and Special Cases

We return to the description of the shortest blocking IS-flow method (SBFM) for solving the maximum IS-flow problem in a network $N = (G = (V, E), u)$ begun in Section 6. Let f be a current IS-flow in N. We show that the task of finding a shortest blocking IS-flow g in (G^+, u_f) (step (P1) of a big iteration of the SBFM) is reduced to finding a totally blocking IS-flow in an acyclic network.

We build the split-graph $\Gamma = S(G^+, u_f)$ and apply the above shortest regular path algorithm to this Γ with the unit length function ℓ on its arcs in order to obtain ϕ, Φ, ξ as in Theorems 7.2 and 7.3. This takes $O(m)$ time (since $L = 1$). Recall that the SPA simultaneously constructs the trimmed 0-graph $\overline{\Gamma}^0$, the main object we will deal with. By Lemma 7.4, $\overline{\Gamma}^0$ is acyclic. The following property is important for us.

Lemma 8.1 *Let $a \in E^+$ be an arc with $u_f(a) > 1$, and let a_1, a_2 be the corresponding split-arcs in Γ. Then $\ell^\xi_\pi(a_1) = \ell^\xi_\pi(a_2)$. Moreover, none of a_1, a_2 can be the base or anti-base arc of any fragment in Φ.*

Proof. Since a_1, a_2 are parallel arcs, for each $\phi \in \Phi$, a_1 enters (resp. leaves) V_ϕ if and only if a_2 enters (resp. leaves) V_ϕ. This implies that $\ell^\xi_\pi(a_1) \neq \ell^\xi_\pi(a_2)$ can happen only if one of a_1, a_2 is the base or anti-base arc of some fragment in Φ. Suppose $a_1 \in \{e_\phi, e'_\phi\}$ for some $\phi \in \Phi$ (the case $a_2 \in \{e_\phi, e'_\phi\}$ is similar). Then (A3) and (A4) in Theorem 7.3 show that a_2 cannot be the base or anti-base arc of any fragment in Φ. Therefore, $\chi_{\phi'}(a_2) \le \chi_{\phi'}(a_1)$ for all $\phi' \in \Phi$, yielding $\ell^\xi_\pi(a_2) \le \ell^\xi_\pi(a_1)$. Moreover, the latter inequality is strict because $\chi_\phi(a_2) = -1 < 1 = \chi_\phi(a_1)$ and $\xi(\phi) > 0$. But $\ell^\xi_\pi(a_1) = 0$, by (A5). Thus, $\ell^\xi_\pi(a_2) < 0$, contradicting (7.2). ■

Let $E^0 \subseteq E^+$ be the set of (images of) zero modified-length arcs of Γ. Lemma 8.1 implies that the base arc e_ϕ of each fragment $\phi \in \Phi$ in Γ is generated by an arc $e \in E^0$ with $u_f(e) = 1$. We can identify these e and e_ϕ and consider ϕ as a fragment in G^+ as well. One can see that $\overline{\Gamma}^0$ is precisely the split-graph for $(\overline{H}, \overline{h})$, where $\overline{H} = (\overline{V}, \overline{E}^0)$ is obtained from $H = (V, E^0)$ by trimming the maximal fragments in Φ, and $\overline{h}$ is the restriction of u_f to $\overline{E}^0$.

Based on the property of each fragment to have unit capacity of the base arc, we reduce (P1) to the desired problem, namely:

(B) *Find a totally blocking IS-flow in $(\overline{H}, \overline{h})$.*

We explain why such a reduction works. Suppose we have found a solution $\overline{g}$ to (B), and let Φ' be the set of maximal fragments $\phi \in \Phi$ with $e_\phi \in \mathrm{supp}(\overline{g})$. For each $\phi \in \Phi'$, we have $\overline{g}(e_\phi) = 1$; therefore, exactly one unit of flow goes out of the head of e_ϕ, through an arc a say. We choose (the image of) a well-crossing path $Q = Q_\phi(x)$ as in (A6) of Theorem 7.3 to connect the head of e_ϕ to the tail x of a in the 0-graph on V_ϕ and push a unit of flow through Q and a unit of flow through the symmetric path Q'. This gives an IS-flow g in H. Moreover, g is a shortest blocking IS-flow in (G^+, u_f). Indeed, the choice of well-crossing paths Q ensures that a symmetric decomposition of g consists of shortest u_f-regular paths, whence g is shortest. Also

G^+ cannot contain a $(u_f - g)$-regular s–s' path R of length $g(E^+)/|g|$. For such an R should be a well-crossing path in H (in view of Theorem 7.2), whence the arcs of R occurring in $\overline{H}$ form an $(\overline{h} - \overline{g})$-regular s–s' path in it and, therefore, $\overline{g}$ is not totally blocking.

Since each path $Q_\phi(x)$ is constructed in $O(|V_\phi|)$ time, and the sets V_ϕ of maximal fragments are pairwise disjoint, the reduction to (B) has linear complexity.

Lemma 8.2 *A totally blocking IS-flow in $(\overline{H}, \overline{h})$ can be extended to a shortest blocking IS-flow in (G^+, u_f), in $O(m)$ time.* ∎

Corollary 8.3 *SBFM solves the maximum IS-flow problem in $O(qT(n,m) + qm)$ time, where q is the number of big iterations ($q \le n$) and $T(n,m)$ is the time needed to find a totally blocking IS-flow in an acyclic network with at most n nodes and m arcs.*

Clearly $T(n,m)$ is $O(m^2)$, as a totally blocking flow can be constructed by $O(m)$ applications of the RA. It looks quite plausible that problem (B) can be solved significantly faster. However, at present we do not know any method for (B) whose complexity is smaller than $O(m^2)$.

Next we are interested in special skew-symmetric networks, in particular, those related to matchings (a relationship between IS-flows and matchings is explained in the next section). For the standard max-flow problem there are well-known special cases of networks $N = (G = (V, E), u)$ with n nodes, m arcs and integer capacities for which the number of big iterations of Dinitz' blocking flow method is significantly less than n. Two of them, having important applications, represent the "unit arc capacity" and "unit node capacity" networks. To combine these into one case, for a node $x \in V$, define the *capacity* $u(x)$ of x to be the minimum of values $\sum_{y:(x,y)\in E} u(x,y)$ and $\sum_{y:(y,x)\in E} u(y,x)$, and define

$$\Delta(N) = \sum(u(x) : x \in V - \{s, s'\}).$$

It was shown in [8, 18] that the number q of big iterations of the blocking flow method does not exceed $2\sqrt{\Delta(N)}$. In particular, if all arc capacities are ones, then $q = O(m^{1/2})$, while if all node capacities $u(x)$, $x \ne s, s'$, are ones, then $q = O(n^{1/2})$ (the latter generalizes the case of networks arising from the bipartite matching problem, for which the complexity $O(n^{1/2}m)$ had been shown in [16, 17]).

The arguments as in [8, 18] are applicable to the skew-symmetric case too.

Lemma 8.4 *The number of big iterations of the SBFM is at most $2\sqrt{\Delta(N)}$.*

Proof. After performing $d = \sqrt{\Delta(N)}$ big iterations, the r-distance from s to s' in the network $N' = (G^+, u_f)$ for the current IS-flow f becomes greater than d, by Lemma 6.1. Let f^* be a maximum IS-flow in N, and let g be defined as in the proof of Theorem 3.3. Then g is a feasible IS-flow in N' and $|g| = |f^*| - |f|$. We assert that $|g| \le d$, which immediately implies that the number of the remaining big iterations is at most $d/2$, thus proving the lemma. To see this, take a symmetric decomposition $\mathcal{D}$ of g consisting of elementary flows (P, P', δ) with $\delta = 1$. Let $\mathcal{D}'$ be the family of all s–s' paths in $\mathcal{D}$; then $|\mathcal{D}'| \ge |g|$. It is easy to see that $u_f(x) = u(x)$ holds for each $x \in V - \{s, s'\}$. Since each path $P \in \mathcal{D}'$ contains at least d nodes $x \neq s, s'$ and uses one unit of the capacity of each of such x's, we have $d|\mathcal{D}'| \leq \Delta(N)$. This implies $|g| \leq d$. ∎

9 Applications to B-Matchings

Given an undirected graph $G' = (V', E')$ with no self-loops, a *matching* M is a subset of edges such that no two edges share a node. The *maximum matching problem* is to find a matching M whose cardinality $|M|$ is as large as possible.

A more general problem is the *maximum capacitated b-matching problem (CBMP)* [6, 20]. The input to this problem includes capacities $u'(e) \in \mathbf{Z}_+$ of the edges $e \in E'$ and *supplies* $b(v) \in \mathbf{Z}_+$ of the nodes $v \in V'$. A *b-matching* is a function $h : E' \to \mathbf{Z}_+$ that satisfies the capacity constraints

$$h(e) \leq u'(e) \qquad \text{for all } \ e \in E',$$

and the supply constraints

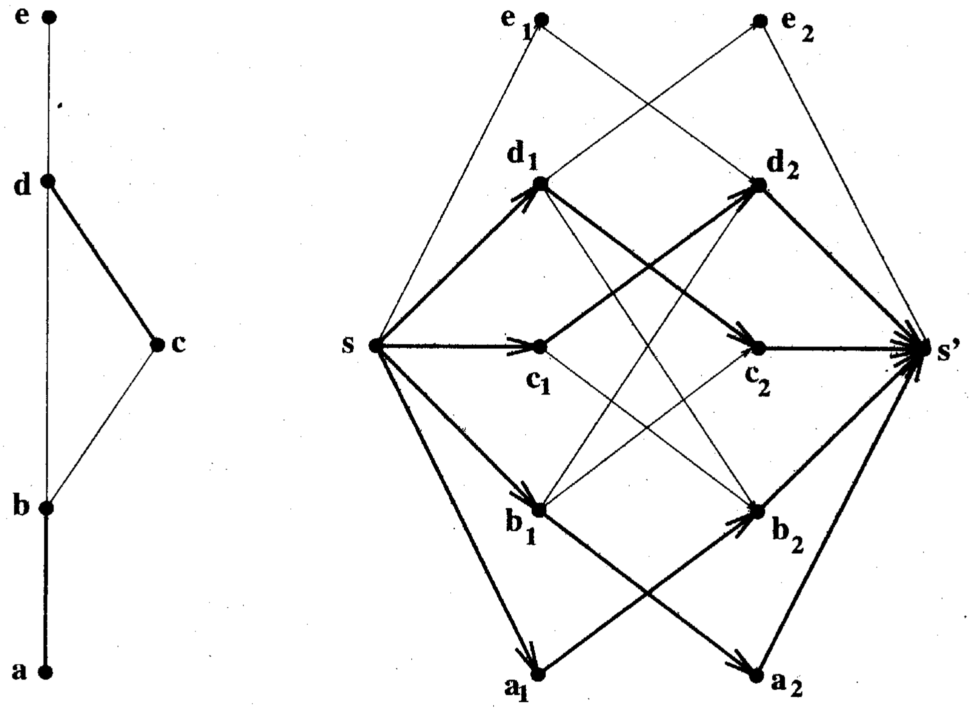
$$\sum_{w:\{v,w\}\in E'} h(v,w) \leq b(v) \qquad \text{for all } \ v \in V'.$$

The *value* of h is $h(E) = \sum_{e \in E'} h(e)$. The goal is to find a b-matching of the maximum value. The maximum matching problem is a special case of the CBMP with all supplies and capacities equal to one. The CBMP is known to be equivalent to the so-called *general matching problem*, the integer linear program in which all entries of the constraint matrix are integers between -2 and $+2$, the sum of the absolute values of entries in each column of the constraint submatrix not including box constraints does not exceed two, and all entries of the objective vector are zeros and ones [6].

The maximum capacitated b-matching problem is reduced to the maximum IS-flow problem without increasing the problem size by more than a constant factor; such a reduction is simple and, in fact, well-known (see, e.g., [1, 2, 19]). More precisely, given an instance of the CBMP, we construct a skew-symmetric digraph $G = (V, E)$ and a symmetric capacity function u on E as follows.

- For each $v \in V'$, V contains two symmetric nodes v_1 and v_2.
- Also V contains two additional symmetric nodes s and s' (the source and the sink).
- For each $\{v, w\} \in E'$, E contains two symmetric arcs (v_1, w_2) and (w_1, v_2) with $u(v_1, w_2) = u(v_2, w_1) = u'(v, w)$.
- For each $v \in V'$, E contains two symmetric arcs (s, v_1) and (v_2, s') with $u(s, v_1) = u(v_2, s') = b(v)$.

Figure 3 illustrates the reduction.



Matching **Symmetric Flow**

Figure 3: *Reduction example for cardinality matching.*

Obviously, there is a one-to-one correspondence between the b-matchings in G' and the IS-flows from s to s' in G, and the value of an IS-flow is twice the value of the corresponding b-matching. Thus, a solution to the MSFP yields a solution to the CBMP, and Theorem 5.1 implies the following result.

Corollary 9.1 *The maximum capacitated b-matching problem can be solved in $O(M(n,m)+nm)$ time, where $M(n,m)$ is the time needed to find an integer maximum flow in a network with n nodes and m arcs.*

Note also that for the case of pure matchings, the corresponding skew-symmetric network has unit capacity at all inner nodes, so the shortest blocking IS-flow method runs in $O(n^{1/2})$ big iterations, by Lemma 8.4.

A slightly different version of the above problem is the *feasible capacitated b-matching problem*: given upper capacities $u'(e)$ and lower capacities $\ell'(e)$ of the edges $e \in E'$, and supplies $b(v)$ and demands $d(v)$ of the nodes $v \in V'$, find a b-matching h satisfying the capacity constraints $\ell'(e) \leq h(e) \leq u'(e)$ and the supply-demand constraints $d(v) \leq \sum_{w:\{v,w\}\in E'} h(v,w) \leq b(v)$. (In particular, this generalizes the well-known degree-constrained subgraph problem where $u' \equiv 1$ and $\ell' \equiv 0$.) The above construction reduces such a problem to the corresponding *feasible IS-flow problem* in which one is asked for finding an IS-flow satisfying lower and upper arc and node capacities. The latter problem is easily reduced to the maximum IS-flow problem in a graph with $O(n)$ nodes and $O(m)$ arcs by a method similar to that for usual flows [9]. This gives the same time bound for the feasible capacitated b-matching problem as in Corollary 9.1.

10 Concluding Remarks

The theoretical results in Sections 3 and 4 have been described in a sketched form in [13]. Subsequent to [13, 14], a similar approach has been investigated in [10, 12, 11]. The cardinality matching algorithm of Blum [2, 3] has a flavor similar to the special case of our shortest blocking IS-flow method on networks with unit node capacities.

As mentioned in Section 8, we have the obvious bound $O(m^2)$ on the complexity of finding a totally blocking IS-flow in an acyclic network, i.e., on $T(n,m)$ in Corollary 8.3. We conjecture that $T(n,m)$ is $O(nm)$ in general case and $O(m)$ in the case of unit arc capacities or unit node capacities, similarly to the corresponding bounds for the blocking flow method.

References

[1] G. M. Adel'son-Vel'ski, E. A. Dinits, and A. V. Karzanov. *Flow Algorithms.* Nauka, Moscow, 1975. In Russian.

[2] N. Blum. A New Approach to Maximum Matching in General Graphs. In *Proc. ICALP*, pages 586–597, 1990.

[3] N. Blum. A New Approach to Maximum Matching in General Graphs. Technical report, Institut für Informatik der Universität Bonn, 1990.

[4] E. A. Dinic. Algorithm for Solution of a Problem of Maximum Flow in Networks with Power Estimation. *Soviet Math. Dokl.*, 11:1277–1280, 1970.

[5] J. Edmonds. Paths, Trees and Flowers. *Canada J. Math.*, 17:449–467, 1965.

[6] J. Edmonds and E. L. Johnson. Matching, a Well-Solved Class of Integer Linear Programs. In R. Guy, H. Haneni, and J. Schönhein, editors, *Combinatorial Structures and Their Applications*, pages 89–92. Gordon and Breach, NY, 1970.

[7] J. Edmonds and R. M. Karp. Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems. *J. Assoc. Comput. Mach.*, 19:248–264, 1972.

[8] S. Even and R. E. Tarjan. Network Flow and Testing Graph Connectivity. *SIAM J. Comput.*, 4:507–518, 1975.

[9] L. R. Ford, Jr. and D. R. Fulkerson. *Flows in Networks.* Princeton Univ. Press, Princeton, NJ, 1962.

[10] C. Fremuth-Paeger and D. Jungnickel. A Unifying Framework for Design and Analysis of Matching Algorithms. Technical Report 379, Institute for Mathematics, University of Augsburg, Germany, 1997.

[11] C. Fremuth-Paeger and D. Jungnickel. Polynomial Augmentation Algorithms for Balanced Network Flows. Technical Report 381, Institute for Mathematics, University of Augsburg, Germany, 1997.

[12] C. Fremuth-Paeger and D. Jungnickel. Simple Augmentation Algorithms for Balanced Network Flows. Technical Report 380, Institute for Mathematics, University of Augsburg, Germany, 1997.

[13] A. V. Goldberg and A. V. Karzanov. Maximum Skew-Symmetric Flows. In *Proc. 3rd European Symp. on Algorithms*, pages 155–170, 1995.

[14] A. V. Goldberg and A. V. Karzanov. Path Problems in Skew-Symmetric Graphs. *Combinatorica*, 16:129–174, 1996.

[15] A. V. Goldberg and S. Rao. Beyond the Flow Decomposition Barrier. In *Proc. 38th IEEE Annual Symposium on Foundations of Computer Science*, pages 2–11, 1997.

[16] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ Algorithm for Maximum Matching in Bipartite Graphs. *SIAM J. Comput.*, 2:225–231, 1973.

[17] A. V. Karzanov. Tochnaya otzenka algoritma nakhojdeniya maksimalnogo potoka, primenennogo k zadache "o predstavitelyakh". In *Problems in Cibernetics*, volume 3, pages 66–70. Nauka, Moscow, 1973. In Russian; title translation: The exact time bound for a maximum flow algorithm applied to the set representatives problem.

[18] A. V. Karzanov. O nakhozhdenii maksimal'nogo potoka v setyakh spetsial'nogo vida i nekotorykh prilozheniyakh. In *Matematicheskie Voprosy Upravleniya Proizvodstvom*, volume 5, pages 81-94. Moscow State University Press, Moscow, 1973. In Russian; title translation: On Finding Maximum Flows in Networks with Special Structure and Some Applications.

[19] E. L. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Reinhart, and Winston, New York, NY., 1976.

[20] L. Lovász and M. D. Plummer. *Matching Theory.* Akadémiai Kiadó, Budapest, 1986.