COMBINATORICA
Akadémiai Kiadó – Springer-Verlag

COMBINATORICA **16** (3) (1996) 353–382

PATH PROBLEMS IN SKEW-SYMMETRIC GRAPHS

ANDREW V. GOLDBERG[1] and ALEXANDER V. KARZANOV[2]

Received November 9, 1993

We study path problems in skew-symmetric graphs. These problems generalize the standard graph reachability and shortest path problems. We establish combinatorial solvability criteria and duality relations for the skew-symmetric path problems and use them to design efficient algorithms for these problems. The algorithms presented are competitive with the fastest algorithms for the standard problems.

1. Introduction

Skew-symmetric graphs come up when certain undirected objects are reduced to directed ones, e.g. [18, 19, 20, 25]. In particular, standard problems on matchings can be stated in terms of such graphs. The present paper is the first in a series of our papers devoted to a theoretical and algorithmic study of problems in skew-symmetric graphs, as well as their applications to matchings. Here we deal with skew-symmetric analogs of the standard reachability and shortest path problems.

By a *skew-symmetric graph* we mean a digraph $G=(V,E)$ for which there is a mapping σ of $V \cup E$ onto itself such that: (i) σ is an involution (i.e., $\sigma(x) \neq x$ and $\sigma(\sigma(x)) = x$); (ii) for every $v \in V$, $\sigma(v) \in V$; (iii) for every $e = (u,v) \in E$, $\sigma(e) = (\sigma(v), \sigma(u))$. We assume that σ is explicitly given and included in the description of G. For brevity we often use the term *symmetric* rather than skew-symmetric. So, we say that node $\sigma(v)$ is symmetric to v, and arc $\sigma(e)$ is symmetric to e. Symmetric elements are also called *mates*. The mate $\sigma(x)$ of an element x will be usually denoted by x'.

Mathematics Subject Classification (1991): 05 C

[1] This research was done while the first author was at Stanford University Computer Science Department, supported in part by ONR Office of Naval Research Young Investigator Award N00014-91-J-1855, NSF Presidential Young Investigator Grant CCR-8858097 with matching funds from AT&T, DEC, and 3M, and a grant from Powell Foundation.

[2] This research was done while the second author was visiting Stanford University Computer Science Department and supported by the above mentioned NSF and Powell Foundation Grants.

Note that if an arc e goes from v to v' then its mate also goes from v to v' (parallel arcs in G are allowed); so the number of arcs (v,v') is even and they are partitioned into pairs of mates.

The symmetry σ is extended to paths (cycles) in a natural way by saying that 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. A path (cycle) P is called *regular*, or an *r-path* (resp. *r-cycle*), if it contains no pair of symmetric arcs; in other words, the arc-sets of P and $P'=\sigma(P)$ are disjoint. In this paper we assume that we are given two distinguished symmetric nodes s and s'. The *regular reachability problem (RRP)* is to find an r-path from s to s' or a proof that there is none. Suppose we are given a *symmetric length function* $\ell: E \to \mathbb{R}$, i.e., $\ell(e)=\ell(e')$ for all $e \in E$. The length $\ell(P)$ of a path P is the sum of the lengths of its arcs. The *shortest regular path problem (SRPP)* is to determine whether (G,ℓ) has a regular cycle of negative length, and if not, find a shortest r-path from s to s'. If ℓ is nonnegative, we refer to SRPP as NSRPP.

The regular reachability problem that we consider is a special case of the forbidden pairs problem in which, given a digraph with two specified nodes s and t and a set of arc pairs, it is required to find a path from s to t which contains none of these pairs. It is known that the latter problem is NP-hard [11].

The aim of this paper is to establish combinatorial solvability and optimality criteria for RRP and SRPP, and to develop fast algorithms for these problems. These problems are at least as hard as the standard reachability and shortest path problems. This is because, given a digraph G with two specified nodes x and y, we can make a disjoint copy G' of G with the opposite orientation of each arc and work with the skew-symmetric graph H formed by adding to $G \cup G'$ new nodes s and s' and arcs $(s,x),(s,y'),(x',s'),(y,s')$. Obviously, any path from s to s' in H is regular and corresponds to a path from x to y in G.

The algorithm for RRP that we develop runs in linear time $O(m)$ (hereinafter $n=|V|$ and $m=|E|$[1]). The algorithm for NSRPP runs in time $O(m \log n)$ or $O(m\sqrt{\log C})$ (the latter bound assumes that lengths are integers in the interval $[0,\ldots,C]$). The corresponding bounds for the usual shortest path problem, achieved by the implementations of Dijkstra's algorithm [5] described in [1] and [10], are $O(m+n\log n)$ and $O(m+n\sqrt{\log C})$, respectively. The algorithm for the general case of SRPP runs in $O(n^2 m \log n)$ time.

This paper is organized as follows. In Section 2 we give a solvability criterion for RRP. The obstruction to the solvability of RRP is somewhat more sophisticated than that for the standard reachability (or connectivity) problem: the existence of a so-called *barrier*, which is analogous to the "Hungarian tree" in matching problems. Using barriers, we then establish an optimality criterion for SRPP in Section 3. Besides node potentials, the criterion involves functions on certain symmetric subgraphs of G, called *fragments*. These results demonstrate nice structural properties of the r-path problems and give rise to combinatorial algorithms for solving

[1] To simplify the presentation, we assume that $m \geq n-1 \geq 2$.

them, described in Sections 4-6. Finally, in Section 7 we discuss relation to matchings.

The alternating and shortest alternating path problems (in a graph with a matching) can be reduced, in linear time, to RRP and SRPP, respectively. The latter problems admit reductions to the former ones in a way similar to that proposed in [26] for the node-regular analog of RRP. However, these reductions considerably increase the problem size. Note also that the above-mentioned solvability and optimality criteria for RRP and SRPP can be derived via these reductions from classical theorems on matchings [3, 6, 7, 24] (see also [21]). The advantage of our direct proofs is that they are given in terms of the input graph itself and based on usual path intuition; they seem to be simpler and more enlightening than the proofs obtained by translating the corresponding results on matchings into the skew-symmetric framework.

In the sequel [14, 15] to this paper we use results obtained here to efficiently solve other natural problems on skew-symmetric graphs, such as the maximum and minimum cost integer symmetric flow problems and their unit capacity variants. Being interesting in their own right, these problems also bridge flows and matchings. The maximum and minimum-cost matching problems and even the "general matching" problem [8] (known also under the name of minimum-cost "bidirected flow" problem) can be reduced to skew-symmetric flow problems while increasing the problem size by only a constant factor.

2. Regular Reachability Problem

2.1. Barriers. Barriers provide infeasibility certificates for RRP. We say that

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

is an *s-barrier*, or, simply, a *barrier*, if the following conditions hold:

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

(B2) $A \cap A' = \emptyset$, where $A' = \sigma(A)$;

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

(B4) For $i = 1, \ldots, k$, there is a unique arc, a_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 $M = V - (A \cup X_1 \cup \ldots \cup X_k)$ and $i = 1, \ldots, k$, no arc connects X_i and M;

(B8) No arc goes from A to $A' \cup M$.

Figure 1 illustrates the definition. Note that arcs from A' to A, from X_i to A, and from M to A are possible. By symmetry, there is no arc from M to A', for each i, there is only one arc, a_i', from X_i to A', and the set M is self-symmetric. If a node x is reachable from s by an r-path, we say that x is r-*reachable*.

We prove the following solvability criterion for RRP.

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

As mentioned in the Introduction, Theorem 2.1 can also be obtained using matching theory. We give a direct and simpler proof of it.

Proof. It is easy to see that s' is not r-reachable if there is a barrier.

Conversely, suppose that no r-path from s to s' exists. We have to prove the existence of a barrier. Let $\langle X\rangle = \langle X\rangle_G$ denote the subgraph of G induced by a set $X \subseteq V$. Consider the set Z of r-reachable nodes in G. Define

$$Z' = \sigma(Z),\ A = Z - Z',\ A' = Z' - Z, X = Z \cap Z', \text{ and } M = V - (Z \cup Z').$$

Let $K_1, \ldots, K_k$ be the weakly connected components of $\langle X\rangle$, and let X_i (E_i) be the node-set (arc-set) of K_i. We call $\mathcal{B} = (A; X_1, \ldots, X_k)$ the *canonical s-barrier*, or, simply, the *canonical barrier*. The theorem is immediatelly implied by the following lemma which shows that $\mathcal{B}$ is a barrier. ∎

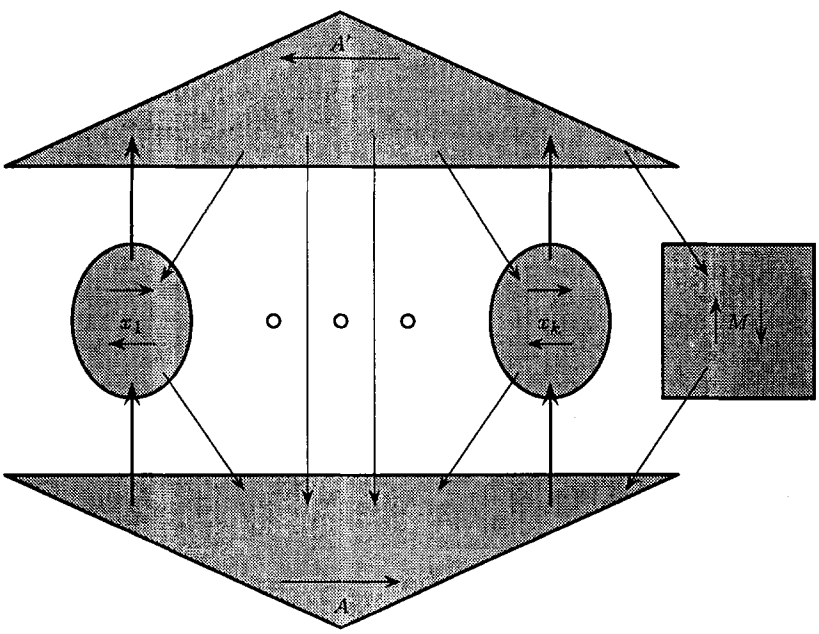

Figure 1. A barrier

Lemma 2.2. *The canonical barrier is indeed a barrier.*

Proof. Let $\mathcal{B} = (A; X_1, \ldots, X_k)$ be the canonical barrier. Properties (B1)–(B2) and (B5)–(B7) are easy. We prove the remaining properties by induction on $|E|$. The base case $|E| = 0$ is trivial.

Let $e_1 = (v_1, w_1), \ldots, e_h = (v_h, w_h)$ be the arcs from A to X. We may assume that $h \geq 2$ (otherwise (B3) and (B4) are trivial). Obviously, there exist distinct

$i,j \in \{1,\ldots,h\}$ such that at least one r-path from s to v_i does not meet the arc e_j; let for definiteness $i=1$ and $j=h$.

Let $\overline{G}=(V,\overline{E})$ be the graph obtained by deleting e_h and e'_h from G. Clearly $\overline{G}$ has no r-path from s to s'. Let $\overline{Z},\overline{Z}',\overline{A},\overline{A}',\overline{X},\overline{M}$ be corresponding sets in the definition of the canonical barrier for $\overline{G}$. The fact that every r-path in $\overline{G}$ is an r-path in G implies that $\overline{Z} \subseteq Z$, whence $\overline{Z}' \subseteq Z'$ and $\overline{X} \subseteq X$. Let $\overline{K}_i = (\overline{X}_i, \overline{E}_i)$, $i = 1,\ldots,p$, be the weakly connected components of the subgraph $\langle \overline{X} \rangle_{\overline{G}}$. By induction $\overline{\mathcal{B}} = (\overline{A}; \overline{X}_1,\ldots,\overline{X}_p)$ is a barrier. Next, the above property for e_1 and e_h yields $v_1, v_h \in \overline{Z}$, whence $v_1, v_h \in \overline{A}$ (as $v_1, v_h \notin X$ and $\overline{X} \subseteq X$). Also $w_1 \in \overline{Z}$. Three cases are possible.

Case 1. $w_h \in \overline{M}$. Let Q be the graph obtained by adding to $\langle \overline{M} \rangle_{\overline{G}}$ the nodes v_h, v'_h and arcs e_h, e'_h. Then Q is skew-symmetric and contains less arcs than G does (as $h \geq 2$ and e_1 is not in Q). Furthermore, e_h is the only arc in Q that leaves v_h, therefore Q has no r-path from v_h to v'_h. By induction the canonical v_h-barrier $(B; Y_1,\ldots,Y_q)$ for Q is a v_h-barrier. One can see that $\widehat{\mathcal{B}} = (\overline{A} \cup B; \overline{X}_1,\ldots,\overline{X}_p, Y_1,\ldots,Y_q)$ is an s-barrier for G. Let $Y = Y_1 \cup \ldots \cup Y_q$ and $\widehat{Z} = \overline{A} \cup B \cup \overline{X} \cup Y$. Since $\widehat{\mathcal{B}}$ is a barrier, each node in $V - \widehat{Z}$ is not r-reachable in G, i.e., $Z \subseteq \widehat{Z}$. On the other hand, each node in $\widehat{Z}$ is obviously r-reachable in G. So $\widehat{Z} = Z$. This easily implies that $\widehat{Z}' = Z'$ and $\widehat{Z} \cap \widehat{Z}' = \overline{X} \cup Y = X$. Obviously, the sets $\overline{X}_1,\ldots,\overline{X}_p, Y_1,\ldots,Y_p$ induce weakly connected components of $\langle \overline{X} \cup Y \rangle_G$. Thus, $\widehat{\mathcal{B}}$ coincides with $\mathcal{B}$.

Case 2. $w_h \in \overline{X} \cup \overline{A}'$. Then, by symmetry, $w'_h \in \overline{X} \cup \overline{A}$. Hence, there is an r-path from s to w'_h in $\overline{G}$. Adding to it the arc e'_h we obtain an r-path from s to v'_h in G. But v'_h is not r-reachable in G; a contradiction.

Case 3. $w_h \in \overline{A}$. Then $w'_h \in \overline{A}'$. Since $w'_h \in Z$, there is an r-path P from s to w'_h in G. Note that P passes through e_h (otherwise v'_h would be r-reachable in G). Let P' be the part of P from w_h to w'_h. Then P' is an r-path in $\overline{G}$ going from $\overline{A}$ to $\overline{A}'$. This contradicts the fact that $\overline{\mathcal{B}}$ is a barrier in $\overline{G}$. ∎

2.2. Bud Trimming. Bud trimming is a kind of equivalent transformation which reduces the graph size and gives an efficient approach to solve RRP (refined in the algorithm for RRP in Section 4). Such a transformation is similar, in a sense, to shrinking a blossom in the classical matching algorithm due to Edmonds [7]. An *s-bud*, or, simply, a *bud*, is a triple $\tau = (\mathcal{V}_\tau, \mathcal{E}_\tau, e_\tau = (v,w))$, where

(D1) $(\mathcal{V}_\tau, \mathcal{E}_\tau)$ is a self-symmetric subgraph of G with $s \notin \mathcal{V}_\tau$;

(D2) e_τ is an arc *entering* $\mathcal{V}_\tau$, i.e., $v \notin \mathcal{V}_\tau \ni w$;

(D3) for every node $x \in \mathcal{V}_\tau$ there is an r-path from w to x in $(\mathcal{V}_\tau, \mathcal{E}_\tau)$ (and, therefore, an r-path from x to w');

(D4) there is an r-path from s to v which does not meet $\mathcal{V}_\tau$.

We refer to e_τ as the *base arc*, and e'_τ as the *anti-base arc* of τ. An important special case of buds arises when the graph $(\mathcal{V}_\tau, \mathcal{E}_\tau)$ is formed by an only r-path Γ from w to w' and its symmetric path Γ'; such a bud is called *elementary*.

The *trimming operation* with respect to τ transforms G into $\overline{G} = (\overline{V}, \overline{E})$ by deleting $\mathcal{E}_\tau$, replacing $\mathcal{V}_\tau$ by two nodes $\overline{w} = \overline{w}_\tau$ and $\overline{w}' = \overline{w}'_\tau$, and replacing the arcs incident to nodes in $\mathcal{V}_\tau$ as follows (see Fig. 2):

(E1) e_τ is replaced by an arc from v to $\overline{w}$, and e'_τ by an arc from $\overline{w}'$ to v';

(E2) each arc $a = (x, y) \neq e'_\tau$ that leaves $\mathcal{V}_\tau$ is replaced by an arc from $\overline{w}$ to y;

(E3) each arc $a = (x, y) \neq e_\tau$ that enters $\mathcal{V}_\tau$ is replaced by an arc from x to $\overline{w}'$;

(E4) each arc $a = (x, y) \notin \mathcal{E}_\tau$ with $x, y \in \mathcal{V}_\tau$ is replaced by an arc from $\overline{w}$ to $\overline{w}'$.

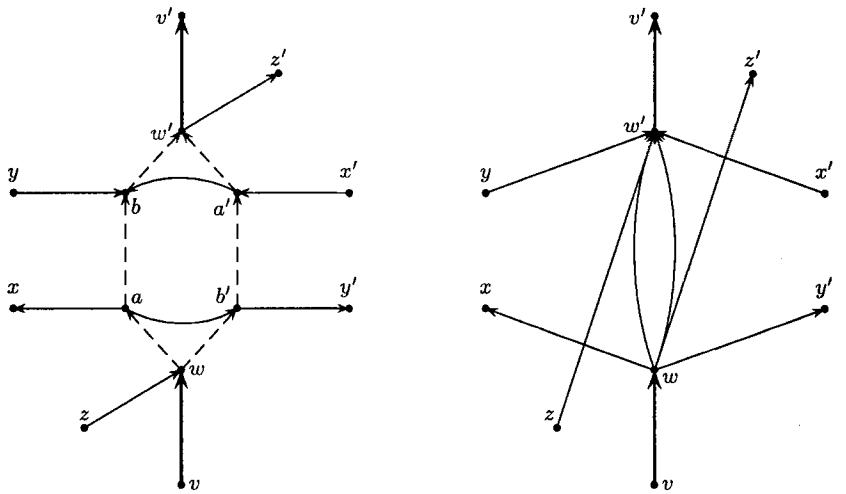

Figure 2. Bud trimming example

For convenience we identify the corresponding arcs in G and $\overline{G}$. We call $\overline{w}$ the *leading* node in the pair $\{\overline{w}, \overline{w}'\}$. Clearly $\overline{G}$ is skew-symmetric in a natural way; this symmetry is denoted by σ as well.

Theorem 2.3. *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 $\overline{G}$.*

Proof. Suppose that s' is not r-reachable in $\overline{G}$. Apply Theorem 2.1 and consider a barrier $\mathcal{B} = (A; X_1, \ldots, X_k)$ for $\overline{G}$. Form the definition of a bud it follows that there

is an r-path P from s to w in G such that e_τ is the last arc of P and w is the only common node of P and $\mathcal{V}_\tau$. Then P corresponds to an r-path from s to $\overline{w}$ in $\overline{G}$. Hence, either (i) $\overline{w} \in X_j$ for some j, or (ii) $\overline{w} \in A$. In case (i), replace $\overline{w}, \overline{w}'$ by $\mathcal{V}_\tau$ in X_j. In case (ii), let I be the set of indices $i \in \{1, \ldots, k\}$ such that the arc a_i from A to X_i has its tail at $\overline{w}$. Remove $\overline{w}$ from A and $\overline{w}'$ from A' and replace in $\mathcal{B}$ the sets X_i $(i \in I)$ by the only set that is the union of these X_i's and $\mathcal{V}_\tau$. It is easy to see that in both cases we obtain a barrier for G. This proves the part "only if".

Conversely, suppose that s' is not r-reachable in G and consider a barrier $\mathcal{B} = (A; X_1, \ldots, X_k)$ for G. The r-reachability from s of all nodes in $\mathcal{V}_\tau$ and the self-symmetry of $\mathcal{V}_\tau$ imply that $\mathcal{V}_\tau \subseteq X_i$ for some i. Then

$$(A; X_1, \ldots, X_{i-1}, (X_i - \mathcal{V}_\tau) \cup \{\overline{w}, \overline{w}'\}, X_{i+1}, \ldots, X_k)$$

is a barrier in $\overline{G}$. ∎

Remark 2.4. Theorem 2.3 remains valid if we use a slightly different trimming operation. Namely, we do not add the arcs from $\overline{w}$ to $\overline{w}'$ as in (E4). In fact, this alternative kind of trimming operation is used in the proof of Theorem 3.2 and in the algorithms for SRPP given in Sections 5 and 6.

In next sections we will encounter the typical situation when a sequence of bud trimming operations is repeatedly applied to the initial graph G. A node u of the resulting graph is called *ordinary* if it is a node of G, and *non-ordinary* otherwise. If u is ordinary, its *preimage* (in G) is just u. If u is non-ordinary, the set of its preimages consists of all nodes of G that were replaced in the subsequence of trimming operations that resulted in appearance of u; in particular, u and u' have the same set of preimages.

The algorithm designed in Section 4 will be based on the following two lemmas. The first lemma is obvious, while the second one is easily proved by arguing as in the proof of the part "only if" in Theorem 2.3.

Lemma 2.5. *Let $\overline{P} = (v_0, e_1, v_1, \ldots, e_k, v_k)$ be an r-path from s to s' in $\overline{G}$.*

(i) *If $\overline{P}$ contains neither $\overline{w}$ nor $\overline{w}'$, then $\overline{P}$ is an r-path in G.*

(ii) *If $v_i = \overline{w}$ for some i, then $e_i = e_\tau$, and the concatenation of $(v_0, e_1, v_1, \ldots, e_{i-1}, v_{i-1}, e_i, w)$, Q and $(x, e_{i+1}, v_{i+1}, \ldots, e_k, v_k)$ is an r-path from s to s' in G, where x is the tail of e_{i+1} in G and Q is an r-path from w to x in $(\mathcal{V}_\tau, \mathcal{E}_\tau)$.* ∎

(If $\overline{P}$ meets $\overline{w}'$ the statement is "symmetric" to that in (ii).)

Lemma 2.6. *Let $\overline{Y} \subset \overline{V}$ be a set such that $s' \notin \overline{Y}$ and $\overline{Y}$ contains all r-reachable nodes in $\overline{G}$. Let Y be the set of all preimages of elements of $\overline{Y}$. Then $s' \notin Y$ and Y contains all r-reachable nodes in G. If, in addition, each element of $\overline{Y}$ is r-reachable in $\overline{G}$ then the same is true for Y and G.* ∎

3. Shortest Regular Path Problem

Let $\ell: E \to \mathbb{R}$ be a symmetric length function. We say that (G,ℓ) is *conservative* (*r-conservative*) if the length $\ell(C)$ of every cycle (resp. every r-cycle) is nonnegative. For $x,y \in V$ let $\mathrm{dist}_\ell(x,y)$ ($\mathrm{rdist}_\ell(x,y)$) denote the *distance* (resp. *regular distance*, or r-*distance*) from x to y, i.e., the minimum length $\ell(P)$ of a path (resp. r-path) from x to y. In these terms, SRPP consists in determining if (G,ℓ) is r-conservative, and if is, finding $\mathrm{rdist}_\ell(s,s')$.

Note that $\mathrm{dist}_\ell(s,s') \le \mathrm{rdist}_\ell(s,s')$, that this inequality may be strict, and that (G,ℓ) may be r-conservative but not conservative. Our aim is to show that if (G,ℓ) is r-conservative then there exists a transformation of ℓ which preserves the r-distance from s to s' and makes it equal to the usual distance from s to s'. Such a transformation involves so-called fragments in G, which are somewhat like buds defined in the previous section.

A *fragment* is a pair $\tau = (V_\tau, e_\tau = (v,w))$, where V_τ is a self-symmetric set of nodes and e_τ is an arc entering V_τ. For instance, every bud corresponds to the fragment defined by its node set and base arc. The set of fragments in G is denoted by $\mathcal{T}$.

For $X \subseteq V$, we denote the set of arcs of G entering (leaving) V by $\delta^+(X)$ (resp. $\delta^-(X)$), and the set $\delta^+(X) \cup \delta^-(X)$ by $\delta(X)$ (the "cut" induced by X). We associate with $\tau \in \mathcal{T}$ its *characteristic function* χ_τ on E defined by

$$\begin{aligned} \chi_\tau(a) &= 1 \quad \text{if } a = e_\tau, e'_\tau, \\ &= -1 \quad \text{if } a \in \delta(V_\tau) - \{e_\tau, e'_\tau\}, \\ &= 0 \quad \text{otherwise.} \end{aligned}$$

Obviously, $\chi_C \cdot \chi_\tau \le 0$ holds for any C that is an r-cycle or an r-path connecting symmetric nodes, where χ_C denotes the incidence vector of C (i.e., $\chi_C(e)$ is 1 if $e \in C$, and 0 otherwise) and ab stands for the inner product of vectors a,b. Therefore, for $\varepsilon \in \mathbb{R}_+$, adding $\varepsilon\chi_\tau$ to ℓ cannot increase the length of C, and the following is true.

Proposition 3.1. *For $\xi : \mathcal{T} \to \mathbb{R}_+$, the function $\ell' = \ell + \sum_{\tau \in \mathcal{T}} \xi(\tau)\chi_\tau$ satisfies $\ell'(C) \le \ell(C)$ for any r-cycle C and any r-path C from s to s'.* ∎

We call ℓ' a *variation* of ℓ . Our main theorem on SRPP is as follows.

Theorem 3.2. *Let s' be r-reachable. Then (G,ℓ) is r-conservative if and only if there exists a variation ℓ' of ℓ such that (G,ℓ') is conservative and*

$$\mathrm{rdist}_\ell(s,s') = \mathrm{rdist}_{\ell'}(s,s') = \mathrm{dist}_{\ell'}(s,s'). \tag{3.1}$$

Proof. The part "if" follows from Proposition 3.1. The part "only if" is more involved. Consider an r-conservative (G,ℓ) and assume that s' is r-reachable. Then

$|\mathrm{rdist}_\ell(s,s')| < \infty$. It suffices to consider a particular case of (G,ℓ), as follows. As before, for a node (arc, path, subset) its symmetric object is denoted with prime.

Without loss of generality, we make the following assumptions.

(i) We assume that G contains a pair of symmetric arcs from s' to s of length $-\mathrm{rdist}_\ell(s,s')$. Adding such arcs to G obviously preserves both the r-conservativity and the r-distance from s to s'.

(ii) Let $N=\{e\in E : \ell(e)<0\}$. We assume that for each $e\in N$, the arcs $e=(x,y)$ and $e'=(y',x')$ are parallel and e,e' are the only arcs leaving x. For if not, we can delete e,e' and add new nodes t,t', arcs $(x,t),(y',t),(t',y),(t',x')$ of zero length and two symmetric arcs from t to t' of length $\ell(e)$. This preserves the r-conservativity and $\mathrm{rdist}(s,s')$. Moreover, given a variation as required in the theorem for the graph constructed this way, it is easy to transform it into the desired variation for the original (G,ℓ) (we leave this to the reader as an exercise).

We show the existence of a variation ℓ' of ℓ such that (G,ℓ') is conservative. Then assumption (i) will immediately derive (3.1).

Let $N=\{h_1,h_1',\dots,h_q,h_q'\}$. We may assume that $q\geq 1$ (for if $N=\emptyset$, $\ell'=\ell$ is as required). Given functions $\pi: V\to\mathbb{R}$ (of node *potentials*) and $\xi:\mathcal{T}\to\mathbb{R}_+$, define the *reduced length* and *modified length* of an arc $e=(x,y)$ to be

$$\ell_\pi(e)=\ell(e)+\pi_x-\pi_y \quad\text{and}\quad \ell_\pi^\xi(e)=\ell_\pi(e)+\sum(\xi(\tau)\chi_\tau \,:\, \tau\in\mathcal{T}), \tag{3.2}$$

respectively. Any potential transformation $\ell\to\ell_\pi$ does not change the length of any cycle. Therefore, it suffices to prove that the program:

$$\text{minimize}\quad \psi(\pi,\xi)=\sum_{i=1}^{q}\max\{0,-2^{q-i}\ell_{\pi,\xi}(h_i)\}\quad\text{subject to} \tag{3.3}$$

$$\pi: V\to\mathbb{R},\quad \xi:\mathcal{T}\to\mathbb{R}_+; \tag{3.4}$$

$$\pi \text{ is } \textit{anti-symmetric}, \text{ i.e., } \pi_v=-\pi_{v'} \text{ for all } v\in V; \tag{3.5}$$

$$\ell_\pi^\xi(e)\geq 0 \quad \text{for all } e\in E-N; \tag{3.6}$$

has a feasible solution (π,ξ) with $\psi(\pi,\xi)=0$. In the proof, we are forced to deal with a sharper form of this program by imposing the following condition on the *support* $\mathcal{T}(\xi)=\{\tau\in\mathcal{T}:\xi(\tau)\neq 0\}$ of ξ:

(3.7) the sets V_τ, $\tau\in\mathcal{T}(\xi)$, form a *nested family*, i.e., for any distinct $\tau,\tau'\in\mathcal{T}(\xi)$, either $V_\tau\subset V_{\tau'}$ or $V_{\tau'}\subset V_\tau$ or $V_\tau\cap V_{\tau'}=\emptyset$;

and the following condition on each $\tau=(V_\tau,e_\tau=(v_\tau,w_\tau))\in\mathcal{T}(\xi)$:

(3.8) any $x\in V_\tau$ is reachable from w_τ by use of a (simple) r-path $P_{\tau,x}$ in $\langle V_\tau\rangle$ such that all arcs on $P_{\tau,x}$ have zero modified length; moreover, if we add to $P_{\tau,x}$ the arc e_τ and an arbitrary arc $(x,y)\in\delta^+(V_\tau)-\{e_\tau'\}$ then the resulting path P satisfies $\chi_P\cdot\chi_{\tau'}=0$ for all $\tau'\in\mathcal{T}(\xi)$ with $V_{\tau'}\subseteq V_\tau$.

Clearly the solution set of {(3.4)–(3.8)} is the union of solution sets of a finite number of linear systems, each defined by a corresponding subset $\mathcal{T}'$ of fragments and a corresponding set of paths $P_{\tau,x}$ for $\tau \in \mathcal{T}'$ and $x \in V_\tau$. Together with the fact that ψ is convex and nonnegative, this implies that {(3.3)–(3.8)} has an optimal solution (π,ξ).

Note that (3.5) and the symmetry of χ_τ for $\tau \in \mathcal{T}$ imply that ℓ_π^ξ is symmetric. Let J be the set of arcs $h \in N$ with $\ell_\pi^\xi(h) < 0$. If $J = \emptyset$, we are done. So assume that $J \neq \emptyset$.

Let $\overline{\mathcal{T}}$ be the set of $\tau \in \mathcal{T}(\xi)$ with V_τ maximal; then $V_\tau \cap V_{\tau'} = \emptyset$ for distinct $\tau, \tau' \in \overline{\mathcal{T}}$, by (3.7). We transform G into $\overline{G} = (\overline{V}, \overline{E})$ in a way similar to bud trimming as in Remark 2.4. More precisely, for each $\tau \in \overline{\mathcal{T}}$, we replace V_τ by two nodes $\overline{w}_\tau, \overline{w}'_\tau$, replace the head of e_τ by $\overline{w}_\tau$, replace the tail of e'_τ by $\overline{w}'_\tau$, and replace the head (tail) of each arc in $E - \{e_\tau, e'_\tau\}$ entering (resp. leaving) V_τ by $\overline{w}'_\tau$ (resp. $\overline{w}_\tau$). For convenience the image in $\overline{G}$ of an arc e of G, if it exists, is denoted by $\overline{e}$; we assign the length $\overline{\ell}(\overline{e})$ to be $\ell_\pi^\xi(e)$. Let $E^0 = \{\overline{e} \in \overline{E} : \overline{\ell}(\overline{e}) = 0\}$; then E^0 is self-symmetric.

Consider the arc $h = h_{i^0} \in J$ with the minimum index i^0. By assumption (ii) both h, h' go from a node t to t'. Moreover, since h, h' are the only arcs entering t', we deduce from (3.8) that $t, t' \notin V_\tau$ for any $\tau \in \mathcal{T}(\xi)$. Hence, $t, t', \overline{h}, \overline{h}'$ are in $\overline{G}$.

We first observe that $G^0 = (\overline{V}, E^0)$ has no r-path from t' to t. Indeed, suppose that such a path $\overline{P} = (t' = x_0, \overline{e}_1, x_1, \ldots, \overline{e}_k, x_k = t)$ exists. Then we can construct an r-path from t' to t in G by replacing each non-ordinary node $x_i = \overline{w}_\tau$ in $\overline{P}$ by the path $P_{\tau,x}$ as in (3.8), where x is the tail of e_{i+1} in G; and doing a similar transformation for each non-ordinary $x_i = \overline{w}'_\tau$ by use of the corresponding symmetric path $P'_{\tau,x}$. Adding h to P, we obtain an r-cycle C in G such that

$$\ell_\pi^\xi(C) = \ell_\pi^\xi(h) + \sum(\ell_\pi^\xi(e) : e \in C - \{h\}) = \ell_\pi^\xi(h) < 0.$$

Furthermore, (3.8) implies $\chi_C \cdot \chi_\tau = 0$ for all $\tau \in \mathcal{T}(\xi)$. Hence,

$$0 > \ell_\pi^\xi(C) = \ell_\pi(C) + \sum(\xi(\tau)\chi_C \cdot \chi_\tau : \tau \in \mathcal{T}) = \ell_\pi(C) = \ell(C),$$

contradicting the r-conservativity of (G, ℓ).

Now our aim is to show that (π,ξ) can be improved so as to give a feasible solution to {(3.3)–(3.8)} with a smaller ψ, thus coming to a contradiction. Consider the canonical t'-barrier $\overline{\mathcal{B}} = (\overline{A}; \overline{X}_1, \ldots, \overline{X}_k)$ in G^0 (possibly $k = 0$). Each $\overline{X}_i$ induces a fragment $\tau_i = (V_{\tau_i}, e_{\tau_i})$ in G in a natural way. Here V_{τ_i} is the set of preimages of elements of $\overline{X}_i$ and e_{τ_i} is the preimage of the arc a_i from $\overline{A}$ to $\overline{X}_i$ in G^0. Let $\mathcal{T}^{\text{new}} = \{\tau_1, \ldots, \tau_k\}$. Let $\mathcal{T}^+$ and $\mathcal{T}^-$ consist of $\tau \in \overline{\mathcal{T}}$ such that $\overline{w}_\tau \in \overline{A}$ and $\overline{w}'_\tau \in \overline{A}$,

respectively. Let $\widetilde{\mathcal{T}} = \mathcal{T}^{\text{new}} \cup \mathcal{T}^+ \cup \mathcal{T}^-$. Let A be the set of ordinary nodes in $\overline{A}$. Define

$$X = \cup(V_\tau : \tau \in \widetilde{\mathcal{T}}) \quad \text{and} \quad M = V - (A \cup A' \cup X).$$

Claim. *There exists $\varepsilon > 0$ such that $\pi' : V \to \mathbb{R}$, defined by*

$$\begin{aligned} \pi'_x &= \pi_x - \varepsilon \quad \text{for } x \in A, \\ &= \pi_x \quad \text{for } x \in X \cup M, \\ &= \pi_x + \varepsilon \quad \text{for } x \in A', \end{aligned} \tag{3.9}$$

and $\xi' : \mathcal{T} \to \mathbb{R}$, defined by

$$\begin{aligned} \xi'(\tau) &= \xi(\tau) + \varepsilon \quad \text{for } \tau \in \mathcal{T}^{\text{new}} \cup \mathcal{T}^+, \\ &= \xi(\tau) - \varepsilon \quad \text{for } \tau \in \mathcal{T}^-, \\ &= \xi(\tau) \quad \textit{otherwise}, \end{aligned} \tag{3.10}$$

satisfy: (i) $\xi'(\tau) \geq 0$ for all $\tau \in \mathcal{T}$, and (ii) $\ell^{\xi'}_{\pi'}(e) \geq 0$ for all $e \in E - J$.

Proof. By (3.10), property (i) holds for any ε between 0 and $\min\{\xi(\tau) : \tau \in \mathcal{T}^-\} > 0$. To obtain (ii), it suffices to examine the edges $e = (x, y) \in E$ such that $\ell^{\xi}_{\pi}(e) = 0$ and the nodes x, y are in different sets among A, A', M and V_τ ($\tau \in \widetilde{\mathcal{T}}$). Consider such an e and an arbitrary $\varepsilon > 0$. Let $p(e) = (\pi'_x - \pi'_y) - (\pi_x - \pi_y)$ and $q(e) = \sum_{\tau \in \mathcal{T}} (\xi'(\tau) - \xi(\tau))\chi_\tau(e)$. Then $\ell^{\xi'}_{\pi'}(e) = \ell^{\xi}_{\pi}(e) + p(e) + q(e)$. We show that $p(e) + q(e) \geq 0$, whence (ii) follows.

The fact that $\mathcal{B}$ is a barrier in G^0 and the construction of A, X imply that the case when $x \in A$ and $y \in A' \cup M$ or $x \in X$ and $y \in M$ is impossible. If $x \in A' \cup M$ and $y \in A$, then (3.9) and (3.10) imply that $p(e) \in \{\varepsilon, 2\varepsilon\}$ and $q(e) = 0$, and we are done. Consider the remaining cases, up to symmetry. Let $\overline{x}$ ($\overline{y}$) denote the tail (resp. head) of the image $\overline{e}$ of e in $\overline{G}$.

Case 1. Let $x \in A$ and $y \in V_\tau$ for some $\tau \in \widetilde{\mathcal{T}}$. Then $p(e) = -\varepsilon$ and $q(e) \in \{\varepsilon, -\varepsilon\}$. We show that $q(e) = \varepsilon$ (and therefore, $p(e) + q(e) = 0$). Obviously, $\overline{x} \in \overline{A}$. Suppose that $\tau \in \mathcal{T}^+$. Since $\overline{y} \in \overline{A}'$ is impossible (as $\overline{\mathcal{B}}$ is a barrier in G^0), we have $\overline{y} = w_\tau$. Hence, $e = e_\tau$. This implies $\chi_\tau(e) = 1$ and $q(e) = \varepsilon$ (by (3.10)). Similarly, if $\tau \in \mathcal{T}^-$ then $\overline{y} \notin \overline{A}'$ implies $\overline{y} = \overline{w}'_\tau$, whence $\chi_\tau(e) = -1$ and $q(e) = \varepsilon$. Finally, if $\tau = \tau_i \in \mathcal{T}^{\text{new}}$ then $\overline{y} \in \overline{X}_i$; so $e = e_{\tau_i}$, $\chi_{\tau_i}(e) = 1$ and $q(e) = \varepsilon$.

Case 2. Let $x \in V_\tau$ and $y \in V_{\tau'}$ for distinct $\tau, \tau' \in \widetilde{\mathcal{T}}$. Then $p(e) = 0$ and $q(e) \in \{2\varepsilon, 0, -2\varepsilon\}$. Suppose that $q(e) = -2\varepsilon$. Then $\chi_\tau(e) = -1$ if $\tau \in \mathcal{T}^+ \cup \mathcal{T}^{\text{new}}$, and $\chi_\tau(e) = 1$ if $\tau \in \mathcal{T}^-$; and similarly for τ'. One can see that this property for e and τ is possible only in the situations as follows: (a) $\overline{x} = \overline{w}_\tau \in \overline{A}$ if $\tau \in \mathcal{T}^+$; (b)

$\overline{x} = \overline{w}'_\tau \in \overline{A}$ if $\tau \in \mathcal{T}^-$; and (c) $\overline{x} \in \overline{X}_i$ if $\tau = \tau_i \in \mathcal{T}^{\text{new}}$. Similarly, $\overline{y} = \overline{w}'_{\tau'} \in \overline{A}'$ if $\tau' \in \mathcal{T}^+$; $\overline{y} = \overline{w}_{\tau'} \in \overline{A}'$ if $\tau' \in \mathcal{T}^-$; and $\overline{y} \in \overline{X}_j$ if $\tau' = \tau_j \in \mathcal{T}^{\text{new}}$. In all cases we get a contradiction with the fact that $\overline{\mathcal{B}}$ is a barrier. ∎

Consider ε, π', ξ' as in the Claim, assuming that $0 < 2\varepsilon \le -\ell^\xi_\pi(h)$ (for $h = h_{i^0}$ as above). Then (π', ξ') satisfies (3.4)–(3.6). Furthermore, for this (π', ξ'), (3.7) is obvious, while (3.8) follows from the construction of G^0 and the fact that for $i = 1, \ldots, k$, each node in $\overline{X}_i$ is reachable from the head of a_i by an r-path in $\langle \overline{X}_i \rangle_{G^0}$. Next, the ends of each $h_j \in J$ do not meet V_τ for $\tau \in \overline{\mathcal{T}}$ (by (3.8)). Hence, $q(h_j) = 0$ and $p(h_j) \ge -2\varepsilon$, where $p(e)$ and $q(e)$ are defined as in the proof of the Claim. We have

$$\ell^{\xi'}_{\pi'}(h_j) = \ell^\xi_\pi(h_j) + p(h_j) \ge \ell^\xi_\pi(h_j) - 2\varepsilon.$$

Furthermore, $0 \le \ell^{\xi'}_{\pi'}(h) = \ell^\xi_\pi(h) + 2\varepsilon$. Now the minimality of i^0 implies

$$\psi(\pi', \xi') - \psi(\pi, \xi) \le -2^{q-i^0}(2\varepsilon) - \sum (2^{q-j}(-2\varepsilon) : h_j \in J - \{h_{i^0}\})$$

$$\le -2 \cdot 2^{q-i^0} \varepsilon (1 - 2^{-1} - 2^{-2} - \ldots - 2^{-q}) < 0.$$

This contradicts the optimality of (π, ξ) and proves the theorem. ∎

Proposition 3.1 and Theorem 3.2 enable us to establish a special duality relation for SRPP, as follows.

Corollary 3.3. *The linear program*

(3.11)
$$\begin{aligned} &\text{maximize} \quad \pi_{s'} - \pi_s \quad \text{subject to} \\ &\pi : V \to \mathbb{R}, \quad \xi : \mathcal{T} \to \mathbb{R}_+, \\ &\pi_x = -\pi_{x'} \quad \text{for } x \in V, \\ &\pi_y - \pi_x - \sum_{\mathcal{T}} \xi(\tau)\chi_\tau(e) \le \ell(e) \quad \text{for } e = (x, y) \in E \end{aligned}$$

has a feasible solution if and only if (G, ℓ) is r-conservative. If (G, ℓ) is r-conservative and s' is r-reachable then (3.11) has a (finite) optimal solution (π, ξ), and vice versa. If (π, ξ) is an optimal solution to (3.11), then $\text{rdist}_\ell(s, s') = \pi_{s'} - \pi_s$.

Proof. The only fact that needs to be explained is that if (G, ℓ) is r-conservative but s' is not r-reachable then (3.11) has a feasible solution (π, ξ) with $\pi_{s'} - \pi_s$ arbitrarily large. Obviously, for any sufficiently large positive number p, adding to G symmetric arcs e, e' from s to s' with $\ell(e) = \ell(e') = p$ preserves the property of being r-conservative. Now s' becomes r-reachable and $\text{rdist}_\ell(s, s') = p$, whence the result follows. ∎

Thus, we may think of (3.11) as the program dual of SRPP. A routine verification shows that an s to s' r-path P and a feasible solution (π, ξ) to (3.11) are

optimal for SRPP and (3.11), respectively, if and only if the following "complementary slackness conditions" hold:

(CS1) for $\tau \in \mathcal{T}$, $\xi(\tau) > 0$ implies $\chi_P \cdot \chi_\tau = 0$;

(CS2) for each arc e in P, $\ell_\pi^\xi(e) = 0$.

In Sections 5 and 6 we describe algorithms for the nonnegative and general versions of SRPP. As a by-product, these algorithms yield the half-integrality property for (3.11) (which is important for purposes of [14, 15] where we deal with flow problems in skew-symmetric graphs).

Theorem 3.4. *If ℓ is integer-valued and (3.11) has an optimal solution then it has a* half-integral *optimal solution.* ∎

4. Regular reachability algorithm

We assume that each skew-symmetric graph we deal with is represented using doubly-linked lists of the outgoing arcs for its nodes. This enables us to access the incoming arcs of a node v as well because, by symmetry, these arcs are the mates of the outgoing arcs of $v' = \sigma(v)$. The algorithm we develop uses bud trimming operations and is based on Lemmas 2.5 and 2.6. As before, we identify the corresponding arcs of the initial and current graphs.

In the current skew-symmetric graph $G = (V, E)$, we grow a set $T \subset E$ which forms a *tree rooted at* s and its node-set A satisfies

$$A \cap A' = \emptyset. \tag{4.1}$$

For convenience we do not distinguish T from its induced graph and refer to T as a *tree*. Initially, $T = \emptyset$, and $A = \{s\}$. For $v \in A$, the path from s to v in T is denoted by P_v, and the path from v' to s' in the symmetric tree T' rooted to s' is denoted by $P'_{v'}$. By (4.1) P_v and $P'_{v'}$ are regular.

At each iteration, the algorithm scans an arc $e = (x, y)$ with $x \in A$ and $y \notin A$. Three cases are possible.

Case 1. Let $y \notin A'$. Then e is added to T. This augments A by y and maintains (4.1).

Now suppose $y \in A'$. Then we trace, step by step, P_x backward from x to s and P'_y forward from y to s', intermixing backward and forward steps. We stop as soon as we discover an arc $a = (v, w)$ in P_x or an arc $a' = (w', v')$ in P'_y such that its mate occurs in the portion of the other path that we have already traced.

Case 2. No such a or a' is discovered. Then P_x and P'_y are completely traced and for each arc of P_x, its mate is not in P'_y. So, the concatenation $\widetilde{P}$ of P_x, e and P'_y

gives an r-path from s to s' in the current graph. The algorithm terminates (the procedure of restoring an s to s' r-path in the initial graph will be described later).

Case 3. An arc as above, say $a=(v,w)\in P_x$, is discovered. Let Q be the part of P_x from w to x and let R be the part of P'_y from y to w'. Then the concatenation P of Q, e and R is an r-path, and $\tau=(\mathcal{V}_\tau,\mathcal{E}_\tau,e_\tau=a)$ is an elementary bud in the current graph, where $(\mathcal{V}_\tau,\mathcal{E}_\tau)$ is the subgraph formed by P and P'. We trim τ as described in (E1)-(E4) in Section 2. The resulting graph $\overline{G}$ becomes the new current graph G. The tree T is updated accordingly; namely, the nodes in Q and R' merge into one node $\overline{w}=\overline{w}_\tau$. Note that updating G consists in forming outgoing arc lists for the new nodes $\overline{w}$ and $\overline{w}'$. The list for $\overline{w}$ is created by concatenating the lists of nodes in $Q\cup R\cup Q'\cup R'$ from which a' and the elements of $\mathcal{E}_\tau$ are removed. The list for $\overline{w}$ consists of the only arc a'.

If the current graph has no arc leaving A, the algorithm declares that s' is not r-reachable. Moreover, in view of Lemma 2.6, the preimages of nodes of A in the initial G are exactly the nodes r-reachable from s; thus, from A one can construct, in linear time, the canonical barrier in G.

The algorithm terminates in finite time. This follows from the facts that trimming a bud τ decreases the number of arcs of the graph by $|\mathcal{E}_\tau|\geq 2$ and that the tree T increases between two consecutive trimmings.

Linear implementation. We show that the algorithm can be implemented to run in time $O(m)$. Let $G=(V,E)$ denote the initial graph, and $\widetilde{G}=(\widetilde{V},\widetilde{E})$ denote the current graph.

The implementation maintains a doubly-linked list L of the arcs of $\widetilde{G}$ that leave A. Initially, L consists of the arcs going out of s. At an iteration, we choose an arbitrary element $e=(x,y)\in L$ to scan. If Case 1 occurs, we examine the arcs q leaving y and add to L each q that does not enter A. Then we delete from L the arcs h that enter y. Since we can efficiently access the outgoing arcs only, in order to find the h's we examine the outgoing arc list of y' and extract the arcs in it that enter A'. We delete the mates of the latter arcs (which are in L). In Case 3, we delete from L the arcs e,e', examine the arcs f leaving nodes in $R\cup Q'$ and add to L each f that neither belongs to $\mathcal{E}_\tau\cup\{a'\}$ nor enters $A-(Q\cup R')$ (such an f turns into an arc leaving A out of $\overline{w}_\tau$ for the new graph and tree). In Cases 1 and 3, each time we add to L an arc from A to A' (for the new graph and tree), we should add its mate too if it is still not in L.

It is easy to check that updating L this way maintains L correctly. To account for the number of operations for maintaining L, note that, in Case 1, the node y is ordinary and the number of arcs we examine is the sum of outdegrees of y and y'. In Case 3, the number of arcs examined is the sum of outdegrees of ordinary nodes in $R\cup Q'$ plus $O(|\mathcal{E}_\tau|)$ (as each non-ordinary node in $R\cup Q'$ obviously has a unique outgoing arc). One can see that, during the algorithm, each ordinary node z can be considered, for the purpose of the examination of its outgoing arcs, at most twice

(once in Case 1 and once in Case 3); in particular, each arc of G can be included in L at most twice. This implies that total time to maintain L and scan arcs is $O(m)$ plus the time needed to recognize the heads in current graphs of the arcs we examine or scan (recall that the arcs are expicitly supported by the adjacency structure of the initial graph G but not of the current graph $\widetilde{G}$).

Next, if Case 3 occurs, the number of arcs that we trace is $O(|\mathcal{E}_\tau|)$ (due to intermixing forward and backward steps). The number of operations involved in trimming τ (concatenating the corresponding lists, and etc.) and transforming T is also $O(|\mathcal{E}_\tau|)$. Since trimming τ decreases the number of arcs of the current graph by $|\mathcal{E}_\tau|$, the total number of operations to perform finding and trimming all buds during the algorithm is $O(m)$.

Finally, when examining or scanning an arc e, we have to recognize efficiently the head x of e in the current graph. Such a task is trivial if x is ordinary. In general situation, we are forced to use certain additional data structures.

We need some terminology and notation. Let u be a non-ordinary node of $\widetilde{G}$. The set of preimages (in G) of u is denoted by $V(u)$; we have $V(u)=V(u')$. The pair $\{u,u'\}$ is considered as a *representative* (in $\widetilde{G}$) of each element of $V(u)$. If u is an ordinary node, then $V(u)$ is $\{u\}$ and the representative of u is just u. The different sets $V(u)$ $(u\in\widetilde{V})$ give a partition of V.

We address the following issue: given a node x of G, how to find its representative in $\widetilde{G}$? Note that if, say, we are interested in recognizing in $\widetilde{G}$ the head $\widetilde{x}$ of an arc $e=(z,x)\in E$ and if we know that the representative of x is $\{u,u'\}$, then $\widetilde{x}\in\{u,u'\}$, and to decide which of u,u' is $\widetilde{x}$ is easy: $\widetilde{x}=u$ if e is the base arc of the corresponding maximal bud, and $\widetilde{x}=u'$ otherwise, assuming that u is the leading node in $\{u,u'\}$ (see the definition in Section 2).

The implementation resolving this issue applies set operations MAKE-SET(x), UNION(x,y), and FIND-SET(x), applicable to a family of pairwise disjoint sets. These operations create a single element set containing x, form the union of the sets containing x and y, and return the name of the set containing x, respectively. In our case, x is a node of G which is a preimage of an element of A, the set containing x is the set $V(u)$ such that $x\in V(u)$, and the representative (u or $\{u,u'\}$) of x is considered as the name of $V(u)$.

From arguments above it follows that the algorithm uses $O(m)$ set operations. In particular, UNION applies only if a bud τ is being trimmed (in Case 3). In this case we go along paths Q and R and merge the sets $V(u)$ for their nodes u into one set, adding elements u' if u is ordinary.

The above operations are supported by the well-known *disjoint set union* (d.s.u.) data structure (see, e.g., [4]). It has a simple implementation ensuring that a sequence of k operations on elements from a universe of size n to be performed in time $O(k\log n+n)$ (which therefore gives $O(m\log n)$ time for our RRP algorithm). The fastest known implementation of the d.s.u. data structure [23]

takes $O(k\alpha(k,n)+n)$ time, where α is the functional inverse of Ackermann's function. Hence, under this implementation the algorithm takes $O(m\alpha(m,n))$ time.

It turns out that we can obtain better running time by observing that in our case the set operations fall into the *incremental tree set union* (i.t.s.u.) version due to Gabow and Tarjan [12]. The i.t.s.u. case is as follows. It deals with a rooted tree F whose nodes represent disjoint sets. Initially, F consists of a single node corresponding to a single element set. The operations allowed on F are contraction of an edge of R, which results in merging the two sets corresponding to its end nodes, and addition of a leaf node corresponding to adding a new element to the ground set. As shown in [12], for this case k operations can be performed in time $O(k+n)$.

In our situation F is T. Indeed, if the algorithm adds a node x to T, this node becomes a leaf and corresponds to a single element set $\{x\}$ in V (as x is obviously an ordinary node of $\widetilde{G}$). If the algorithm trims a bud τ by use of paths Q and R (in Case 3), then the transformation of T can be interpreted as a sequence of contractions of the edges in $Q \cup R'$ (we should also add to T as leaves the ordinary nodes in $Q' \cup R$ and then contract their incident edges as well).

Thus, application of the i.t.s.u. techniques gives time $O(m)$ for the algorithm described above. It either declares the problem infeasible, or finds an s to s' r-path $\widetilde{P}$ in the last graph $\widetilde{G}$. We now explain how to transform, in liner time, this $\widetilde{P}$ to the desired r-path in the initial G.

Extracting the path. This relies on bookkeeping and postprocessing added to the above algorithm.

A sequence $(a_1,\ldots,a_k)$ of arcs of the initial G is called a *partial path* if there is a path $(b_1,\ldots,b_r)$ in G and a sequence of indices $i_1<i_2<\ldots<i_k$ such that $a_j=b_{i_j}$ for $1\le j\le k$ (considering a path as an arc set).

For the bookkeeping, we store information about all trimmed buds in a way which enables us to efficiently undo the trimmings during the postprocessing. We maintain a rooted forest $\mathcal{F}$ that represents the nested structure of the buds which appeared during the algorithm. Each vertex of $\mathcal{F}$ is either a node of the initial G or a pair $\{\overline{w}_\tau,\overline{w}'_\tau\}$ created by trimming a bud τ at some iteration of the algorithm. The leaves of $\mathcal{F}$ are exactly the nodes of G. For each non-leaf vertex $u=\{\overline{w}_\tau,\overline{w}'_\tau\}$, the leaves of the subtree of $\mathcal{F}$ rooted at u are the preimages of $\overline{w}_\tau$. Obviously, the maximal trees in $\mathcal{F}$ correspond to the maximal buds and the ordinary nodes of the last graph.

The forest $\mathcal{F}$ is maintained as follows. At the beginning, $\mathcal{F}$ is the trivial forest with n vertices and no edges. When we trim a bud τ, we add to $\mathcal{F}$ a new vertex $u=\{\overline{w}_\tau,\overline{w}'_\tau\}$ and make the roots of the corresponding trees in $\mathcal{F}$ children of u. We also store the paths P_τ, P'_τ from w_τ to w'_τ that form $(\mathcal{V}_\tau,\mathcal{E}_\tau)$ in the current graph. These paths are stored as the corresponding sequences of arcs of the initial G, organized as doubly-linked lists. Clearly, bookkeeping is performed in time $O(m)$.

The postprocessing stage restores the required path in G. The forest $\mathcal{F}$ is changed during the stage by removing (repeatedly) certain roots. We use the following F-PATH(x) operation: given a leaf x of $\mathcal{F}$, find the path $S(x)$ connecting x with the root of the tree in $\mathcal{F}$ that contains x. Once such an operation applied to x, the x is labelled as *active*. This means that the path $S(x)$ already exists and can therefore be used on further iterations without applying F-PATH operation again. Initially, each leaf is non-active.

The initial path $\widetilde{P}$ in $\widetilde{G}$ corresponds to a partial path P in G. If P is a path in G, we are done. If not, P has consecutive arcs $e=(a,b)$ and $h=(c,d)$ such that $b\neq c$. In this case b and c are covered by a bud, and therefore, belong to the same tree in $\mathcal{F}$. We examine the paths $S(b)$ and $S(c)$ (after constructing $S(q)$ for $q=b,c$ by use of F-PATH operation if needed) and extract the common part $\widehat{S}$ of $S(b)$ and $S(c)$. Clearly the last vertex of $\widehat{S}$ corresponds to the minimal bud τ that covers both b and c. We remove $\widehat{S}$ from $\mathcal{F}$ and accordingly remove the part $\widehat{S}$ from $S(b)$ and $S(c)$. (The new $\mathcal{F}, S(b), S(c)$ concern the graph $\widehat{G}$ obtained from $\widetilde{G}$ if we undo trimming the buds corresponding to the vertices in $\widehat{S}$.)

Obviously, either e is the base arc $e_\tau=(\mathcal{V}_\tau, w_\tau)$ of the τ, or h is its anti-base arc $e'_\tau=(w'_\tau, v'_\tau)$. Assume the former; the other case is symmetric. Our aim is to find a path $\Gamma=(e_1,\ldots,e_t)$ in $(\mathcal{V}_\tau,\mathcal{E}_\tau)$ to be inserted in P between e and h (in order to get an r-path in $\widehat{G}$). To this purpose, we examine the paths P_τ and P'_τ. Let y be the first vertex in the new path $S(c)$. Two cases are possible.

1) y is a node of G, i.e., $y=c$. Then c is a node of P_τ or P'_τ; say, c is in P_τ. The desired Γ is the part of P_τ from w_τ to c.

2) y is a pair $\{\overline{w}_{\tau'}, \overline{w}'_{\tau'}\}$. Obviously, $\mathcal{E}_\tau$ contains the base and anti-base arcs of the bud τ'. Since h leaves $\mathcal{V}_\tau$, the only case is possible when the tail of h in $\widehat{G}$ is $\overline{w}_{\tau'}$. Hence, we should take as Γ the path from w_τ to $\overline{w}_{\tau'}$ in $(\mathcal{V}_\tau,\mathcal{E}_\tau)$, i.e., Γ is the part from w_τ to $\overline{w}_{\tau'}$ of P_τ or P'_τ.

We repeat the procedure for the new current graph $\widetilde{G}:=\widehat{G}$ and path P, and so on. As a result, we eventually find the desired path in G.

Formally, at an iteration of the postprocessing stage we deal with a partial path $(a_1,\ldots,a_k)$ in G, scan a pair (a_i,a_{i+1}) in it, and decide whether the head b of a_i coincides the tail c of a_{i+1}. If $b\neq c$, we find a path $\Gamma=(e_1,\ldots,e_r)$ in $(\mathcal{V}_\tau,\mathcal{E}_\tau)$ to insert between a_i and a_{i+1} as described above, where τ is the minimal bud that covers both b,c. Clearly, the scan of all the pairs (a_i,a_{i+1}) throughout the algorithm can be organized so as to take linear time. Determining the required path Γ takes time $O(|\mathcal{E}_\tau|)$. Finally, the total time to perform all necessary F-PATH operations is bounded by the size of the initial forest $\mathcal{F}$, which is $O(m)$.

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

Theorem 4.1. *The RRP algorithm can be implemented so that it solves RRP in linear time.* ∎

5. Shortest regular path algorithm under nonnegative lengths

The algorithm for SRPP combines ideas behind the proof of Theorem 3.2 and technical tools used in the algorithm for RRP. In this section we consider the problem with a *nonnegative* symmetric length function ℓ, for which the algorithm is simpler, and refer to this problem as NSRPP. We use terminology and notations from Sections 3 and 4 and assume that s' is r-reachable (from s). Once again, we emphasize that any fragment $\tau=(V_\tau,e_\tau)$ we deal with is defined in the initial graph, whereas any bud $\tau=(\mathcal{V}_\tau,\mathcal{E}_\tau,e_\tau)$ is in the current graph.

The algorithm maintains anti-symmetric potentials π on V and a nonnegative function ξ on a subset $\mathcal{T}'$ of fragments so that the modified length function $\ell'=\ell_\pi^\xi$ (cf. (3.2)) is nonnegative (assuming that ξ is extended by zero on the fragments not in $\mathcal{T}'$). Moreover, we assume that the sets V_τ for $\tau\in\mathcal{T}'$ form a nested family (cf. (3.7)), and that each fragment $\tau=(V_\tau,e_\tau=(v_\tau,w_\tau))$ in $\mathcal{T}'$ satisfies (3.8) (with $\mathcal{T}'$ instead of $\mathcal{T}(\xi)$) and the following conditions:

(5.1) $s\notin V_\tau$;

(5.2) $\ell'(e_\tau)=0$;

(5.3) v_τ is r-reachable from s by an r-path of zero modified length which is disjoint from V_τ.

Define $\overline{\mathcal{T}}$ to be the set of $\tau\in\mathcal{T}'$ with V_τ maximal and define graphs $\overline{G}=(\overline{V},\overline{E})$ and $G^0=(\overline{V},E^0)$ as in the proof of Theorem 3.2. As before, we identify the corresponding arcs in $\overline{G}$ and G. The algorithm grows in G^0 a tree $T\subset E^0$ rooted at s. The node set A spanned by T satisfies $A\cap A'=\emptyset$ and

(5.4) for each $\tau\in\overline{\mathcal{T}}$, the node $\overline{w}_\tau$ is contained in A.

Initially, $\pi=0$, $\xi=0$, $T=\emptyset$, and $A=\{s\}$. At each iteration, the algorithm searches for an arc $e=(x,y)\in E^0$ with $x\in A$ and $y\notin A$. Suppose that such an arc exists. Then the algorithm proceeds as the RRP algorithm with respect to $\overline{G}$. More precisely, three cases are possible.

Case 1. If $y\notin A'$, e is added to T.

Otherwise we trace the path P_x in T and the path P'_y in T'.

Case 2. Let the concatenation $\widetilde{P}$ of P_x, e and P'_y be regular. Using the path extraction procedure as in the RRP algorithm, we transform $\widetilde{P}$ into an s to s' r-path P in G such that each arc of P has zero modified length. So, (CS2) is satisfied. Moreover, (3.8) provides (CS1). Therefore, $\ell(P)=\pi(s')-\pi(s)$, i.e., P is a shortest r-path. The algorithm terminates.

Case 3. Let Q be the part of P_x and R be the part of P'_y such that the concatenation P of Q, e and R is an r-path and the arc a in P_x preceding Q is the mate of the arc in P'_y following R. Then $P\cup P'$ and a form a bud in G^0, and $\tau=(V_\tau,e_\tau)$ forms

a fragment in G, where $e_\tau = a$ and V_τ is the set of preimages in G of nodes from $P \cup P'$. We put $\xi(\tau)=0$ and add τ to $\overline{\mathcal{T}}$. This corresponds to updating $\overline{G}$ by use of the trimming operation (as in Remark 2.4) with respect to $\overline{G}$ and the above bud. Accordingly, G^0 and T are updated. One can see that (3.7), (3.8) and (5.1)–(5.4) continue to hold.

It remains to consider the following case.

Case 4. G^0 has no arc from A to $\overline{V}-A$. Then $\ell'(e)>0$ for all arcs from A to $\overline{V}-A$ in $\overline{G}$ (the set of these arcs is nonempty since s' is r-reachable). Let $M=\overline{V}-(A\cup A')$ (all nodes in M are ordinary, by (5.4)). For $X,Y \subset \overline{V}$ let (X,Y) denote the set of arcs from X to Y in $\overline{G}$. Define

$$\varepsilon_1 = \min\{\ell'(e) \,:\, e \in (A,M)\}, \qquad \varepsilon_2 = \min\{\ell'(e) \,:\, e \in (A,A')\}, \tag{5.5}$$
$$\text{and} \quad \varepsilon = \min\{\varepsilon_1, \varepsilon_2/2\}.$$

Next, let $\widetilde{A}$ be the set of ordinary nodes in A. The algorithm updates π and ξ as follows:

$$\begin{aligned} \pi(x) &:= \pi(x) - \varepsilon \quad \text{for } x \in \widetilde{A}, \\ &:= \pi(x) + \varepsilon \quad \text{for } x \in \widetilde{A}'; \\ \xi(\tau) &:= \xi(\tau) + \varepsilon \quad \text{for } \tau \in \overline{\mathcal{T}} \end{aligned} \tag{5.6}$$

(preserving π and ξ on the other elements). Arguing as in the proof of the Claim in Section 3, one can see that updating (5.6) changes ℓ' as follows:

$$\begin{aligned} \ell'(e) &:= \ell'(e) - \varepsilon \quad \text{for } e \in (A,M) \cup (M,A'), \\ &:= \ell'(e) - 2\varepsilon \quad \text{for } e \in (A,A'), \\ &:= \ell'(e) + \varepsilon \quad \text{for } e \in (M,A) \cup (A',M), \\ &:= \ell'(e) + 2\varepsilon \quad \text{for } e \in (A',A), \end{aligned} \tag{5.7}$$

This implies that the tree T remains correct and properties (3.8), (5.2), (5.3) continue to hold. The definition of ε preserves nonnegativity of ℓ'. Moreover, the transformation of ℓ' results in appearance of an arc e from A to $\overline{V}-A$ in $\overline{G}$ for which $\ell'(e)=0$. Hence, at the next iteration, one of Cases 1-3 will occur. This implies that the algorithm terminates (with Case 2) in finite time and solves NSRPP.

Efficient implementation. We show that the algorithm can be implemented to run in time close to linear. The only nonlinear term comes from $O(m)$ operations on priority queue data structures. In fact, it suffices to design fast implementations for searching for an arc of zero modified length from A to $\overline{V}-A$, updating π, and computing the minima in (5.5) (the other operations take linear time in total). The implementation does not explicitly maintain the arc-set of G^0.

1) The potentials π of the preimages of nodes in $\overline{V}-(\widetilde{A}\cup\widetilde{A}')$ are stored explicitly. For each $x\in\widetilde{A}$, a number $p(x)$ is stored and its potential is given implicitly in the form $\pi(x)=p(x)+d$, where d is a number common for all nodes in $\widetilde{A}$. Then $\pi(x)$ for $x\in A'$ is quickly computed, when needed, as $\pi(x)=-p(x')-d$. If a node y is added to $\widetilde{A}$ (in Case 1), we set $p(y)$ equal to $\pi(y)-d$. As soon as a node x is removed from $\widetilde{A}$ (in Case 3), we set its potential to $\pi(x)=p(x)+d$. Updating $\pi(x)$ for all $x\in\widetilde{A}$ (in Case 4) is performed implicitly by decreasing d by ε. Also we maintain the forest $\mathcal{F}$ as in the implementation of the RRP algorithm, which represent the nested structure of fragments in $\mathcal{T}'$. We assume that for $\tau\in\mathcal{T}'$ the number $\xi(\tau)$ is attached to the node $\{\overline{w}_\tau,\overline{w}_{\tau'}\}$ of $\mathcal{F}$.

2) We assume that for each fragment τ in $\overline{\mathcal{T}}$ none of the arcs e of $\langle V_\tau\rangle_G$ was deleted when constructing $\overline{G}$, i.e., such an e is replaced by an arc from $\overline{w}_\tau$ to $\overline{w}'_\tau$ (so there is a one-to-one correspondence between the arcs in G and $\overline{G}$). The reason for modifying the definition of $\overline{G}$ this way is to avoid extractions and deletions of such arcs when trimming the corresponding bud. At the same time, this modification does not affect correctness of the above algorithm because, once we meet an arc from $\overline{w}_\tau$ to $\overline{w}'_\tau$ when choosing an appropriate arc leaving A, we can simply ignore this arc. If a bud τ' in $\overline{G}$ is trimmed, the outgoing arc list for the new node $\overline{w}_{\tau'}$ is formed by concatenating the lists of the nodes of this bud from which the only arc $e'_{\tau'}$ is deleted.

3) The arcs of $\overline{G}$ (and therefore, of G) are partitioned into three parts E^1,E^2 and E^3. The set E^1 consists of the arcs with both ends either in A or in A' or in M, and the values of ℓ' on these arcs are maintained explicitly. The set E^3 consists of the arcs from $\overline{w}_\tau$ to $\overline{w}'_\tau$ for $\tau\in\overline{\mathcal{T}}$, and E^2 is the rest. Next, the set $E^2\cup E^3$ is further partitioned into four self-symmetric sets $(A,M)\cup(M,A')$, (A,A'), $(M,A)\cup(A',M)$ and (A',A) denoted by L_1,L_2,L_3,L_4, respectively; clearly $E^3\subseteq L_2$. Note that under the transformation of ℓ' in Case 4 (cf. (5.7)) the values of ℓ' are not changed on $E^1\cup E^3$ and are changed by adding $-\varepsilon,-2\varepsilon,\varepsilon,2\varepsilon$ on the arcs in L_1,L_2-E^3,L_3,L_4, respectively. We maintain ℓ' on E^2 implicitly as follows. For $i=1$, 2, 3, 4 and $e\in L_i$, a number $\lambda(e)$ is stored, and a number D_i common for the members of L_i is maintained so that if $e\in E^2$ then the current $\ell'(e)$ equals $\lambda(e)+D_i$. Note that the arcs e in E^3 (which are within L^2) are also provided automatically with $\lambda(e)$ because, before getting in E^3, an arc e has already occurred in L_2 (as being a member of E^2 which becomes a member of E^3 under trimming in Case 3). The modified length of $e\in E^3$ is, in general, different from $\lambda(e)+D_2$ and, in fact, not maintained. Initially, all D_i's are zero, each L_i is initialized by the corresponding arcs e incident to s or s', and $\lambda(e)=\ell(e)$.

4) In Cases 1 and 3, appropriate arcs are added to or deleted from L_i's. If an arc is moved from L_i to L_j, we regard this as a deletion from L_i followed by an

insertion into L_j. If e is deleted from L_i we set $\ell'(e)$ to $\lambda(e)+D_i$. If e is added to L_i we store $\lambda(e)$ to be $\ell'(e)-D_i$. (So, transferring e from L_i to L_j involves updating $\lambda(e):=\lambda(e)+D_i-D_j$.) Each time an arc is added to (deleted from) L_i, we do the same with its mate. By this rule, one can transform the sets L_i using merely the outgoing arc lists of ordinary nodes in $\overline{G}$.

More precisely, in Case 1, we examine the arcs going out of y and y'. The arcs in $(y,M-y'),(y,A'),(y,y'),(y,A)$ are added to L_1, transferred from L_1 to L_2, added to L_2, deleted from L_3, respectively. The arcs in $(y',A'),(y',M-y),(y',A),(y',y)$ are deleted from L_1, added to L_3, transferred from L_3 to L_4, added to L_4, respectively. (Here we do not distinguish between a one-element set and its only element.) In Case 3, we examine the outgoing arcs of ordinary nodes z' in Z', where Z is the set of nodes in $Q\cup R'$. The sets $(z',A'),(z',Z),(z',M),(z',A-Z)$ are added to L_2, transferred from L_4 to L_2, transferred from L_3 to L_1, deleted from L_4, respectively.

We leave it to the reader to check correctness of the above transformations of the sets L_i. Moreover, one can check that for a fixed L_i, each arc can be included in L_i at most once during the algorithm. Thus, the sets L_i are maintained in linear time.

5) Each set L_i is organized as a priority queue (p.q.) data structure. For $e\in L_i$, the number $\lambda(e)$ is considered as a *key* according to which e is ordered in L_i. Note that all arcs leaving A are contained in $L_1\cup L_2$. At an iteration, we choose a minimal element e_1 in L_1. If $\varepsilon_1:=\lambda(e_1)+D_1=0$ then e_1 (or e_1' if $e_1\in(M,A')$) is just the desired arc e in E^0 and we obtain Case 1. Otherwise we choose a minimal element e_2 of L_2 that is not in E^3. If $\varepsilon_2:=\lambda(e_2)+D_2=0$ then e_2 is the desired e in E^0 and we obtain Case 2 or 3. Finally, we observe that ε_1 and ε_2 are exactly the numbers as in (5.5), assuming that $\varepsilon_1=\infty$ if L_1 is empty and that $\varepsilon_2=\infty$ if L_2 has no element in E^2. Thus, if both ε_1 and ε_2 are non-zero then we obtain Case 4, compute ε as in (5.5) and add $-\varepsilon,-2\varepsilon,\varepsilon,2\varepsilon$ to D_1,D_2,D_3,D_4, respectively (which corresponds to the transformation of ℓ' as in (5.7)).

Note that, once e_1 in L_1 is chosen, e_1 and its mate are deleted from L_1 (as they no longer belong to $(A,M)\cup(M,A')$). In contrast, e_2 as above (as well as each member of E^3) should remain in the new (A,A'). To overcome this difficulty, we may think of L_2 as consisting of two parts, L' and L''. L' contains all current members of $E^2\cap(A,A')$ and some members of E^3, and L'' contains the rest. We assume that only L' is organized as a p.q. data structure and choose a minimal element e just in L', after which e is moved from L' to L'' (in particular, if the minimal element found in L' belongs to E^3, it is immediately moved to L'' and we search for the next minimal element in L').

The operations we execute on a p.q. data structure are ADD(x), DELETE(x) and FIND-MINIMUM. These add a new element x, delete x and find an element with the minimum key. The above analysis shows that the implementation runs in time $O(m)$ plus $O(m)$ priority queue operations (over a set of size $O(m)$). The latter

operations take $O(m \log n)$ time if we use the simple binary heap data structure (see, e.g., [4]), or $O(m\sqrt{\log C})$ if we use the R-heap implementation [1], assuming that the lengths $\ell(e)$ are integers not exceeding C. This gives the following.

Theorem 5.1. *NSRPP can be solved in time* $O(m \min\{\log n, \sqrt{\log C}\})$. ∎

6. Shortest regular path algorithm under arbitrary lengths

The algorithm for SRPP with arbitrary (symmetric) lengths generalizes the above algorithm for NSRPP. As a preprocessing step, we make a *node-splitting transformation* to construct an equivalent problem which is at most a constant factor bigger but has only $O(n)$ negative length arcs. This transformation (repeatedly) replaces a node x that has an outgoing arc of negative length by two nodes x_1 and x_2, replaces the outgoing arcs (x,z) by arcs (x_1,z) with $\ell(x_1,z)=\ell(x,z)-\mu$, and replaces the incoming arcs (z,x) by arcs (z,x_2) with $\ell(z,x_2)=\ell(z,x)-\mu$, where μ is the minimum length of an arc going out of x. Also it adds an arc (x_1,x_2) with $\ell(x_1,x_2)=2\mu$. If x is split this way, so is x'. We observe that the resulting graph is skew-symmetric in a natural way, the new function ℓ is symmetric, and if $\ell(e)<0$ for an arc $e=(u,v)$ then e is the only arc leaving u and the only arc entering v.

Let $e_1,e_1',\ldots,e_N,e_N'$ be the negative length arcs, and for $i=0, 1, \ldots, N$, let $G_i=(V,E_i)$ be the graph obtained from G by deleting the arcs e_j,e_j' for $j=i+1,\ldots,N$. The algorithm consists of $N+1$ *outer iterations*. After $i-1$ outer iterations, there are an (anti-symmetric) potential function π_i on V and a positive function ξ_i on a set $\mathcal{T}_i$ of fragments in G_{i-1} such that $\mathcal{T}_i$ satisfies (3.7) and (3.8) and

(6.1) the modified length $\ell_i(e)=\ell_{\pi_i}^{\xi_i}(e)$ of each arc e of G_{i-1} is nonnegative

(note that (5.2) needs not be satisfied for $\tau\in\mathcal{T}_i$).

Initially, $\mathcal{T}_1=\emptyset$ and $\pi_1=0$. At ith outer iteration, we examine $\ell_i(e_i)$, and if it is nonnegative, we finish the iteration by setting $\pi_{i+1}=\pi_i$, $\mathcal{T}_{i+1}=\mathcal{T}_i$ and $\xi_{i+1}=\xi_i$. Otherwise we form the auxiliary graph $H=(V^H,E^H)$ by adding to G_{i-1} new nodes t and t' and arcs $(t,v),(t,u'),(u,t'),(v',t')$ of zero length ℓ, where $e_i=(u,v)$. Since v has no incoming arc in G_{i-1}, u and v belong to no fragment in $\mathcal{T}_i$ (in view of (3.8)); so we may think that fragments of $\mathcal{T}_i$ were "trimmed" directly in H. We extend $\pi=\pi_i$ to t and t' so that the modified lengths of the added arcs are nonnegative.

An r-path P from t to t' in H is called *strong* with respect to a family $\widetilde{\mathcal{T}}$ of fragments if for any $\tau\in\widetilde{\mathcal{T}}$, $\chi_\tau\cdot\chi_P=0$.

Suppose we have a procedure which transforms π_i, $\mathcal{T}_i$, ξ_i, ℓ_i into π_{i+1}, $\mathcal{T}_{i+1}$, ξ_{i+1}, ℓ_{i+1} satisfying (3.7), (3.8) and (6.1) (for H) and simultaneously finds a strong path P w.r.t. $\mathcal{T}_{i+1}$ with all arcs of zero length ℓ_{i+1}. Since $t,t'\notin V_\tau$ for any $\tau\in\mathcal{T}_{i+1}$

(in view of (3.8)), $\mathcal{T}_{i+1}$ induces a set of fragments in G_i in a natural way (this set is also denoted by $\mathcal{T}_{i+1}$). Moreover, one can see that if we replace in P the nodes t,t' and their incident arcs by e_i or e_i', then we obtain an r-cycle Q in G_i such that $\chi_\tau \cdot \chi_Q = 0$ for any $\tau \in \mathcal{T}_{i+1}$. Compute $a = \ell_{i+1}(e_i)$ $(= \ell_{i+1}(e_i'))$. If $a \geq 0$, then (6.1) holds for G_i (and we go to the next outer iteration). If $a < 0$, we conclude that (G,ℓ) is not r-conservative (since $\ell(Q) = \ell_{i+1}(Q) = a < 0$).

If after executing N outer iterations we found out that (G,ℓ) is r-conservative, then at the last, $(N+1)$th, outer iteration we put H,t,t' to be G,s,s', respectively, and find a strong r-path P from s to s', along with $\pi,\mathcal{T}'$ and ξ. Then (CS1) and (CS2) are satisfied, and therefore, P is a shortest s to s' r-path for (G,ℓ).

Description of ith outer iteration. If $\mathcal{T}_i$ is empty, the iteration is the same as the whole NSRPP algorithm in Section 5. Otherwise, the iteration is similar but somewhat more complicated because for some fragments τ, $\xi(\tau)$ may decrease and if it reaches zero, then τ is deleted from the current $\mathcal{T}_i$; this results in a transformation of the current graph by expanding the corresponding trimmed bud.

We assume that at the beginning of this outer iteration we are given $\pi = \pi_i$, $\mathcal{T}' = \mathcal{T}_i$, $\xi = \xi_i$ (for H) and the graph $\overline{G} = (\overline{V}, \overline{E})$ obtained from H by "trimming" the fragments in $\overline{\mathcal{T}}$ as described in the implementation of the NSRPP algorithm, where $\overline{\mathcal{T}}$ is the set of maximal fragments in $\mathcal{T}'$.

The outer iteration works with current $\pi,\mathcal{T}',\xi,\ell' = \ell_\pi^\xi \geq 0,\overline{\mathcal{T}},\overline{G}$, and grows a tree $T \subset \overline{E}$ in $\overline{G}$ rooted at t such that node-set A of T satisfies $A \cap A' = \emptyset$. It consists of *inner* iterations, each of which searches for an arc e from A to $\overline{V} - A$ with $\ell'(e) = 0$ and deals with one of Cases 1-4. Cases 1, 2 and 3 are the same as in the NSRPP algorithm. If there is no arc e from A to $\overline{V} - A$ with $\ell'(e) = 0$, then we obtain Case 4 and transform π and ξ in the following way, which is motivated by the proof of Theorem 3.2 and is slightly different from the analogous case for NSRPP.

Case 4. Let $\mathcal{T}^+$ $(\mathcal{T}^-)$ be the set of fragments $\tau \in \overline{\mathcal{T}}$ such that $\overline{w}_\tau \in A$ (resp. $\overline{w}'_\tau \in A$). Let $M = \overline{V} - (A \cup A')$ and let $\widetilde{A}$ be the set of ordinary nodes in A; then A is partitioned into $\widetilde{A}$, $\{\overline{w}_\tau : \tau \in \mathcal{T}^+\}$ and $\{\overline{w}'_\tau : \tau \in \mathcal{T}^-\}$. Let Y be the set of edges $e \in \overline{E}$ from $\overline{w}_\tau$ to $\overline{w}'_\tau$ for $\tau \in \overline{\mathcal{T}}$. Define

$$
\begin{aligned}
\varepsilon_1 &= \min\{\ell'(e) \,:\, e \in (A, M)\}, \\
\varepsilon_2 &= \min\{\ell'(e) \,:\, e \in (A, A') - Y\}; \\
\varepsilon_3 &= \min\{\xi(\tau) \,:\, \tau \in \mathcal{T}^-\}; \\
\varepsilon &= \min\{\varepsilon_1, \varepsilon_2/2, \varepsilon_3\}.
\end{aligned} \tag{6.2}
$$

If $\varepsilon < \infty$ (i.e., at least one of the three sets in (6.2) is nonempty), we transform π and ξ as follows:

$$
\begin{aligned}
\pi(x) &:= \pi(x) - \varepsilon \quad \text{for } x \in \widetilde{A}, \\
&:= \pi(x) + \varepsilon \quad \text{for } x \in \widetilde{A}'; \\
\xi(\tau) &:= \xi(\tau) + \varepsilon \quad \text{for } \tau \in \overline{\mathcal{T}}^{+}; \\
&:= \xi(\tau) - \varepsilon \quad \text{for } \tau \in \overline{\mathcal{T}}^{-}
\end{aligned} \tag{6.3}
$$

(preserving π and ξ on the other elements).

An analysis as in the proof of the Claim in Section 3 shows that this transformation changes ℓ' in the same way as in (5.7) (except that, instead of (A, A') and (A', A), one should put $(A, A') - Y$ and $(A', A) - Y$, respectively). Also one can see that (3.8) and (6.1) continue to hold.

If $\varepsilon = \infty$ then we transform π and ξ as in (6.3) with respect to a sufficiently large (finite) positive ε and finish the outer iteration. So, if $i \le N$, the choice of ε ensures that the modified length of e_i becomes nonnegative; whereas if $i = N+1$, the emptiness of both (A, M) and $(A, A') - Y$ implies that s' is not r-reachable from s', and the problem has no solution.

Let $\varepsilon < \infty$. If ε equals ε_1 or $\varepsilon_2/2$ then the inner iteration finishes (and Case 1, 2 or 3 will happen at the next inner iteration). Otherwise a fragment in $\mathcal{T}^{-}$ arises for which the new value of ξ is zero. We continue the inner iteration by choosing one of such fragments (if there are many) and deleting it from $\mathcal{T}'$. This means that the corresponding trimmed bud $\tau = (\mathcal{V}_\tau, \mathcal{E}_\tau, e_\tau)$ is expanded in $\overline{G}$ in place of $\overline{w}_\tau, \overline{w}'_\tau$; in other words, the new $\overline{G}$ is the graph obtained from H by "trimming" the maximal fragments in the new set $\mathcal{T}'$. Accordingly, we update the tree T so that the new tree contains all previous (and, possibly, some additional) arcs. If e is the arc in T entering $\overline{w}'_\tau$, then its new head becomes the corresponding node z in $\mathcal{V}_\tau$. Furthermore, if the old tree has an arc leaving $\overline{w}'_\tau$, then this arc is exactly the anti-base arc e'_τ. In this case we extract the corresponding path from z to w'_τ in $(\mathcal{V}_\tau, \mathcal{E}_\tau)$ and insert it in T between e and e'_τ.

We observe that the outer iteration terminates (with Case 2 or Case 4 when $\varepsilon = \infty$) after executing $O(n)$ inner iterations. Indeed, let Z be the set of preimages in H of nodes in $\widetilde{A} \cup \{\overline{w}_\tau : \tau \in \mathcal{T}^{+}\}$. Let B (C) be the set of fragments $\tau \in \mathcal{T}'$ such that $V_\tau \subseteq V_{\tau'}$ for some $\tau' \in \mathcal{T}^{+}$ (resp. $\tau' \in \overline{\mathcal{T}} - \mathcal{T}^{+}$). One can see that during the outer iteration the sets Z and B are monotone non-decreasing and C is monotone non-increasing. Moreover, Case 3 increases B. In Case 4, if $\varepsilon = \varepsilon_1$ or $\varepsilon_2/2$ then Case 1, 2 or 3 occurs at the next inner iteration; otherwise C decreases. Finally, in Case 1, the node y added to A is either ordinary or $\overline{w}_\tau$ or $\overline{w}'_\tau$ for some $\tau \in \overline{\mathcal{T}}$. In the first and second cases Z increases, and in the third case $\overline{w}'_\tau$ will stay at A until either τ vanishes in Case 4 or $\overline{w}'_\tau$ is included in a bud in Case 3 (so, C will decrease). Since the size of each of Z, B, C is obviously $O(n)$, we conclude that the number of inner iterations is indeed $O(n)$.

Next we show that an outer iteration can be implemented in $O(nm\log n)$ time. This implies the following bound on the algorithm.

Theorem 6.1. *SRPP can be solved in* $O((N+1)nm\log n)$, *or* $O(n^2m\log n)$ *time.*

1) First we explain how to expand a bud τ (in Case 4). Recall that the bud trimming operation for τ (i) replaced a subset $\mathcal{V}_\tau$ of nodes of a certain current graph $\widetilde{G}$ by two nodes $\overline{w}_\tau$ and $\overline{w}'_\tau$, and (ii) formed the outgoing arc list $\mathcal{L}$ for $\overline{w}_\tau$ by concatenating the lists $\mathcal{L}(z)$ of nodes z of $\mathcal{V}_\tau$. We assume that for every $z\in\mathcal{V}_\tau$ two pointers were stored when creating $\mathcal{L}$ which mark the beginning and the end of the sublist $\mathcal{L}(z)$ in $\mathcal{L}$. Thus, to undo (ii), we split $\mathcal{L}$ into $|\mathcal{V}_\tau|$ lists using these pointers, which takes $O(|\mathcal{V}_\tau|)$ time. To undo (i), we need a disjoint set union data structure that allows the SPLIT(r) operation in addition to the MAKE-SET(x), UNION(x,y), and FIND-SET(x) operations (see Section 4). The SPLIT(r) operation, applied to the set containing r formed by UNION(x,y), partitions this set into the original two sets which were combined by the UNION operation. The *union by rank* variant of the d.s.u. data structure (see, e.g., [4]) implements the desired operations so that each operation takes $O(\log n)$ time (where n is the maximum set size). Since the sum of sizes of all buds trimmed or/and expanded during the outer iteration is $O(n)$, it takes $O(n\log n)$ time to execute all these operations.

2) The arcs of $\overline{G}$ connecting different sets among A, A' and M are partitioned into four sets L_1, L_2, L_3, L_4, and their modified lengths are maintained in a similar way as in the implementation of the NSRPP algorithm. Note that an arc from $\overline{w}_\tau$ to $\overline{w}'_\tau$ for $\tau\in\overline{\mathcal{T}}$ may be included in L_2 (if $\tau\in\mathcal{T}^+$) or in L_4 (if $\tau\in\mathcal{T}^-$). There are three essential differences compared with the NSRPP case. The first one is that M may contain non-ordinary nodes. The second one is that each arc may be included in and deleted from a set L_i many times (e.g., this can be easily shown for arcs in L_1). The third one is that when a fragment τ in $\mathcal{T}^-$ is destroyed (in Case 4), some arcs e from $\overline{w}_\tau$ to $\overline{w}'_\tau$ no longer connect the pair of nodes of a trimmed fragment, and therefore, we have to restore their real modified length; however, it is unclear how to do this fast.

3) One can see that each extra deletion (insertion) of an arc from (to) L_i may happen only in Case 4 when some fragment in $\mathcal{T}^-$ is destroyed. Thus, the number of operations applied to each p.q. data structure L_i during an outer iteration is $O(nm)$, whence the total time is $O(nm\log n)$. If $\tau\in\mathcal{T}^-$ is being destroyed, we examine the outgoing arcs e of all nodes of the initial graph H that form the set V_τ and for each e decide where e should be moved to.

4) When $\tau\in\mathcal{T}^-$ is destroyed, we also examine the arcs e from $\overline{w}_\tau$ to $\overline{w}'_\tau$. If both ends u and v of e in H belong to the same fragment $\tau'\in\mathcal{T}'$ with $V_{\tau'}\subset V_\tau$ (i.e., if, as before, e will connect the pair of nodes of a trimmed fragment in the new graph), then we do nothing. Otherwise we compute $\ell'(e)$ as in expression (3.2). To this aim we trace the corresponding paths for u and v in the forest $\mathcal{F}$ to extract values of ξ we need; therefore, computing $\ell'(e)$ takes $O(n)$ time. Since this procedure is applied to each arc at most once, the total time for "deep" restoration

the modified lengths of arcs is $O(nm)$. Note also that this procedure is also applied to the appropriate arcs at the end of each outer iteration in order to get the explicit values of ℓ' in the input of the next outer iteration.

Summing up the above arguments we conclude that an outer iteration can be performed in $O(nm \log n)$ time, thus proving Theorem 6.1.

Remark. Borrowing ideas from [17], one can avoid an arc to be included many times in the same p.q. data structure. It turns out that, instead of maintaining L_1, L_3 and L_4, it suffices to maintain, for each node $x \in V^H$ whose image is in $M \cup \{\overline{w}_\tau, \overline{w}'_\tau : \tau \in \mathcal{T}^-\}$, two sets $L^1(x)$ and $L^2(x)$, each organized as a p.q. data structure. Here $L^1(x)$ ($L^2(x)$) consists of the arcs (x,z) with z to be a preimage of a node in $Z' = \widetilde{A} \cup \{\overline{w}'_\tau : \tau \in \mathcal{T}^+\}$ (resp. in Z). In addition, we maintain certain lists containing nodes x for which $L^1(x)$ or $L^2(x)$ is nonempty. These structures enable us to reduce the total time for choosing appropriate arcs leaving A and computing minima in (6.2) during an outer iteration to $O(n^2 + m \log n)$. Also one can organize restorations of ℓ' on arcs from $\overline{w}_\tau$ to $\overline{w}'_\tau$ diring an outer iteration so that they takes $O(n^2)$ time. This gives time $O(n^3 + nm \log n)$ for the whole algorithm. Unfortunately, limits of the paper do not allow us to come into details of this improved version. To compare: the Bellman-Ford method for the usual shortest path problem with arbitrary lengths runs in time $O(nm)$ [2,9,22], and the scaling algorithm in [13] in time $O(\sqrt{n}m \log N)$, where $-N$ is a lower bound on the arc lengths.

Dual half integrality. In conclusion of this section we show that the algorithm for SRPP implies Theorem 3.4. (This can be also derived from a theorem on dual half-integrality for the weighted perfect matching problem.) It suffices to prove the following lemma.

Lemma 6.2. *If the current functions π and ξ are half-integral and $\pi(t') - \pi(t)$ is an integer before executing the transformation in Case 4, then the number ε in (6.2) is a half-integer.*

Proof. Since ε_1 and ε_3 are obviously half-integral, the lemma will follow from the fact that ε_2 is an integer. To see the latter, consider an arc $e \in (A, A') - Y$. We know that $\ell'(a) = 0$ for all arcs in T and T'. This and (3.8) imply that there is a t to t' path P in H such that P contains e and all other arcs of P have zero modified length. Then

$$\ell'(e) = \ell'(P) = \ell(P) + \pi(t) - \pi(t') + \sum(\xi(\tau)\chi_\tau \cdot \chi_P : \tau \in \mathcal{T}').$$

Since $\ell(P)$ and $\pi(t) - \pi(t')$ are integers, ξ is half-integral and $\chi_\tau \cdot \chi_P$ is even for any $\tau \in \mathcal{T}'$ (as $t, t' \notin V_\tau$), we conclude that $\ell'(e)$ is an integer. ∎

7. Relationship to matchings

As mentioned in the Introduction, RRP can be reduced to a certain matching problem in a way similar to that described in [26] for its node-regular analog. Given a skew-symmetric graph $G=(V,E)$ and symmetric nodes s and s', we form an undirected graph $H=(T,U)$ as follows. The set T consists of t,t' and nodes v_a corresponding to arcs $a\in E$. The set U is $M\cup Y\cup W\cup W'$, where

(i) M consists of edges $\{v_a,v_{a'}\}$ for each pair $\{a,a'\}$ of mates in E;

(ii) Y consists of edges $\{v_a,v_{b'}\}$ for $a,b\in E$ such that the head of a coincides with the tail of b;

(iii) W consists of edges $\{t,v_{a'}\}$ for $a\in E$ such that s is the tail of a;

(iv) W' consists of edges $\{v_a,t'\}$ for $a\in E$ such that s' is the head of a.

Because of the symmetry, pairs $\{a,b\}$ and $\{b',a'\}$ generate parallel edges in (ii); we identify these edges. Clearly M is a matching in H covering all nodes except t and t'. A regular path $P=(x_0,a_1,x_1,\ldots,a_k,x_k)$ from $s=x_0$ to $s'=x_k$ in G corresponds to the path $\Psi(P)$ from t to t' in H given by the sequence $(t,v_{a_1'},v_{a_1},v_{a_2'},\ldots,v_{a_k'},v_{a_k},t)$; moreover, $\Psi(P)$ is *alternating*, i.e., $\Psi(P)$ is simple and each odd or each even edge in it is in M. One can see that Ψ gives a one-to-one correspondence between the simple r-paths from s to s' in G and the alternating paths from t to t' in H. Therefore RRP is equivalent to the problem of finding an alternating path from t to t' in H. From the algorithmic viewpoint, however, this reduction is expensive since $|T|=|E|+2$ and $|U|$ can be significantly larger than $|E|$. We leave it to the interested reader to derive Theorems 2.1, 2.3 and Lemma 2.2 from classic results on matchings.

This reduction is extended to NSRPP as follows. For $e\in U$, define $\mu(e)$ to be zero if $e\in M$, $(\ell(a)+\ell(b))/2$ if $e=\{v_a,v_{b'}\}\in Y$, and $\ell(a)/2$ if $e=\{t,v_{a'}\}\in W$ or $e=\{v_a,t'\}\in W'$. Then for any r-cycle or any t to t' r-path P in G, $\ell(P)$ is equal to the length of $\Psi(P)$ with respect to μ. To solve our problem, we find a minimum weight perfect matching M' for H,μ. Clearly the symmetric difference $M\triangle M'$ consists of a simple path Q containing t and t', and a (possibly empty) collection of pairwise disjoint cycles $C_1,\ldots,C_k$ of zero weight μ. Thus, $P=\Psi^{-1}(Q)$ satisfies $\ell(P)=\mu(Q)=\mu(M')$, and P is a shortest r-path from s to s' in G.

Now consider the general case of SRPP. We first determine whether (G,ℓ) is r-conservative. Connect t and t' by an edge e_0 with a negative length of large absolute value, obtaining $\widetilde{H}$ and $\widetilde{\mu}$. Find a minimum weight perfect matching M' for $\widetilde{H},\widetilde{\mu}$. By the choice of $\widetilde{\mu}(e_0)$, M' must contain e_0. We observe that (G,ℓ) is r-conservative if and only if $M\triangle M'$ has no negative weight cycles. If (G,ℓ) is r-conservative, we proceed in the same way as for nonnegative ℓ. Theorem 3.2 and Corollary 3.3 can be derived (though this is difficult) from classic results of Edmonds on weighted matchings [6].

Next we give applications of RRP and SRPP to matching problems. First we consider an augmenting path problem. Suppose we are given an undirected graph $H=(T,U)$ and a matching M in it, and let $Z \subseteq U$ be the set of nodes not covered by M. We are interested in finding an alternating path connecting two distinct nodes in Z. Such a problem is reduced to RRP for $G=(V,E)$, s, s' as follows. The set V is obtained by splitting each $v \in T$ into two nodes v_1 and v_2, and adding two more nodes, s and s'. The set E consists of the arcs (u_1,v_2) and (v_1,u_2) for each edge $\{u,v\} \in U-M$, the arcs (u_2,v_1) and (v_2,u_1) for each edge $\{u,v\} \in M$, and the arcs (s,v_1) and (v_2,s') for each $v \in Z$. One can see that there is a natural one-to-one correspondence between the alternating paths in H connecting distinct nodes of Z and the r-paths from s to s' in G. Note also that $|V|=O(|T|)$ and $|E|=O(|U|)$, so the algorithm of Section 4 solves the augmenting path problem in linear time.

Now suppose that, in addition to H and M, we are given a weighting $\mu: U \to \mathbb{R}$, and we wish to find a minimum weight alternating path Q connecting distinct nodes of Z, or show that there is an alternating cycle C of negative weight. To solve this problem, for each arc $a \in E$ obtained from an edge $e \in U$, define $\ell(a)=\mu(e)$, and for the other arcs $a \in E$ define $\ell(a)=0$. Then the problem is reduced to SRPP for G,s,s',ℓ.

Consider the following problem: given an undirected graph $H'=(T',U')$ with nonnegative weights on its edges and an element (node or edge) x in it, find a shortest odd cycle going through x. This problem reduces to the shortest augmenting path problem with only $O(1)$ increase in the problem size (see, e.g., [16], Section 8.3.6). A similar reduction applies if we are interested in an even cycle through an edge, or an odd or even path connecting prescribed nodes. Thus, the algorithm in Section 5 solves each of these problems in time $O(|U'| \log |T'|)$.

Acknowledgement. We are thankful to the referees for many useful suggestions intended to improve the presentation style of this paper.

References

[1] R. K. AHUJA, K. MEHLHORN, J. B. ORLIN, and R. E. TARJAN: Faster Algorithms for the Shortest Path Problem, *Technical Report* CS-TR-154-88, Department of Computer Science, Princeton University, 1988.

[2] R. E. BELLMAN: On a Routing Problem, *Quart. Appl. Math.* **16** (1958), 87–90.

[3] C. BERGE: Two Theorems in Graph Theory, *Proc. Nat. Acad. Sci. U.S.A.* **43** (1957), 842–844.

[4] T. H. CORMEN, C. E. LEISERSON, and R. L. RIVEST: *Introduction to Algorithms*, (MIT Press, Cambridge, MA, 1990).

[5] E. W. DIJKSTRA: A Note on Two Problems in Connection with Graphs, *Numer. Math.* **1** (1959), 269–271.

[6] J. EDMONDS: Maximum Matchings and a Polyhedron with 0,1-vertices, *J. Res. Nat. Bur. Stand.* **96B** (1965), 125–130.

[7] J. EDMONDS: Paths, Trees and Flowers, *Canada J. Math.* **17** (1965), 449–467.

[8] 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* (Gordon and Breach, NY, 1970), 89–92.

[9] L. R. FORD, JR., and D. R. FULKERSON: *Flows in Networks* (Princeton Univ. Press, Princeton, NJ, 1962).

[10] M. L. FREDMAN, and R. E. TARJAN: Fibonacci Heaps and Their Uses in Improved Network Optimization Algorithms, *J. Assoc. Comput. Mach.* **34** (1987), 596–615.

[11] H. N. GABOW, S. N. MAHESHWARI, and L. OSTERWEIL: On Two Problems in the Generation of Program Test Paths, *IEEE Trans. on Software Eng.* **SE-2** (1976), 227–231.

[12] H. N. GABOW, and R. E. TARJAN: A Linear-Time Algorithm for a Special Case of Disjoint Set Union, *J. Comp. and Syst. Sci.* **30** (1985), 209–221.

[13] A. V. GOLDBERG: Scaling Algorithms for the Shortest Paths Problem, In: *Proc. 4th ACM-SIAM Symposium on Discrete Algorithms*, 1993, 222–231.

[14] A. V. GOLDBERG, and A. V. KARZANOV: Maximum Flows in Skew-Symmetric Graphs, *Technical Report* (Stanford University, Stanford, 1995).

[15] A. V. GOLDBERG, and A. V. KARZANOV: Minimum Cost Flows in Skew-Symmetric Graphs, in preparation.

[16] M. GRÖTSCHEL, L. LOVÁSZ, and A. SCHRIJVER: *Geometric Algorithms and Combinatorial Optimization* (Springer Verlag, 1988).

[17] A. V. KARZANOV: An algorithm for determining a maximum packing of odd-terminus cuts and its applications, in: *Studies in Applied Graph Theory* (A. S. Alekseev, ed. , Nauka, Novosibirsk,1986), 126–140; in Russian; English translation in *Amer. Math. Soc. Translations*, Ser. 2, **158** (1994), 57–70.

[18] A. V. KARZANOV: A Minimum Cost Maximum Multiflow Problem, In: *Combinatorial Methods for Flow Problems* (Inst. for Systems Studies, Moscow, volume 4, 1979), 138–156; In Russian.

[19] A. V. KARZANOV: Minimum Cost Multiflows in Undirected Networks, *Mathematical Programming* **66** (3) (1994), 313–325.

[20] L. LOVÁSZ: The Factorization of Graphs II, *Acta Math. Hung.* **23** (1972), 223–246.

[21] L. LOVÁSZ, and M. D. PLUMMER: *Matching Theory* (Akadémiai Kiadó, Budapest, 1986).

[22] E. F. MOORE: The Shortest Path Through a Maze, In: *Proc. of the Int. Symp. on the Theory of Switching* (Harvard University Press, 1959), 285–292.

[23] R. E. TARJAN: Efficiency of a Good but not Linear Set Union Algorithm, *J. Assoc. Comput. Mach.* **22** (1975), 1975.

[24] W. T. TUTTE: The Factorization of Linear Graphs, *J. London Math. Soc.* **22** (1947), 107–111.

[25] W. T. TUTTE: Antisymmetrical Digraphs, *Canada J. Math.* **19** (1967), 1101–1117.

[26] H. YINNONE: On Paths Avoiding Forbidden Pairs of Vertices in a Graph, submitted to *Discrete Appl. Math.*

Andrew V. Goldberg

NEC Research Institute, Inc.
4 Independence Way
Princeton, NJ 08540, USA
avg@research.nj.nec.com

Alexander V. Karzanov

Institute for System Analysis
of Russian Acad. Sci.
9, Prospect 60 Let Oktyabrya,
Moscow, Russia
karzanov@cs.vniisi.msk.su