Multilength Single Pair Shortest Disjoint Paths *

Cristina G. Fernandes[1] † Hein van der Holst[2] ‡ José Coelho de Pina[1]

[1] Instituto de Matemática e Estatística
Universidade de São Paulo - Brazil
Rua do Matão 1010, 05508-090 São Paulo/SP, Brazil
E-mail: {cris,coelho}@ime.usp.br

[2] Fachbereich Mathematik und Informatik,
Freie Universität Berlin,
Arnimallee 2–6, D-14195 Berlin, Germany
E-mail: hvdholst@math.fu-berlin.de

Topics: algorithms and computational complexity.

Abstract

The k-SHORTEST PATHS problem consists of: given a digraph D, a pair (s,t) of vertices of D and k non-negative functions $l_1,\dots,l_k$ on the arcs of D, find k internally vertex-disjoint paths $P_1,\dots,P_k$ from s to t such that $l_1(P_1)+\cdots+l_k(P_k)$ is as small as possible. We describe, for each fixed k, a polynomial-time algorithm for the k-SHORTEST PATHS restricted to acyclic digraphs. We prove two complexity results: unless P = NP, for each constant c, there is no polynomial-time n^c-approximation algorithm (1) for the 2-SHORTEST PATHS, where n is the number of vertices of D, and (2) for the k-SHORTEST PATHS restricted to acyclic digraphs. We also show a polynomial-time algorithm for a multicommodity variation of the problem in planar graphs.

*Research supported in part by PRONEX/CNPq 664107/1997-4 (Brazil).
†Research supported in part by CNPq Proc. 301174/97-0 (Brazil).
‡Research supported in part by CAPES-PROAP (Brazil).

1 Introduction

The well-known single pair shortest path problem consists of: given a digraph D, a non-negative function l on the arcs of D and two vertices s and t, find a path P from s to t that minimizes $l(P)$, where $l(P)$ denotes the sum of $l(e)$ over all arcs e in P. This problem is solvable in polynomial time. We address the following generalization of the single pair shortest path problem, which we call k-SHORTEST PATHS:

given : – a digraph $D = (V, A)$;
– a pair (s, t) of vertices of D;
– non-negative functions $l_1, \ldots, l_k$ on the arcs of D;

find : – k internally vertex-disjoint paths $P_1, \ldots, P_k$ from s to t such that

$$l_1(P_1) + \cdots + l_k(P_k)$$

is as small as possible.

For $l_1 = \ldots = l_k$ the k-SHORTEST PATHS reduces to the min-cost flow problem and, therefore, can be solved in polynomial time.

We consider first the problem on acyclic digraphs. Algorithms for finding arc-disjoint paths in acyclic digraphs have applications on scheduling problems [1] and aircraft assignment problems [7]. We reformulate the k-SHORTEST PATHS in acyclic digraphs in terms of finding a shortest path in a (large) acyclic digraph. This is a known reformulation due to Perl and Shiloach [6] for finding two vertex-disjoint paths in an acyclic digraph. Later this was extended by Fortune, Hopcroft and Wyllie [3] in order to derive a polynomial-time algorithm for the k vertex-disjoint paths problem in acyclic digraphs (see also Schrijver [7]). From this reformulation, we derive the theorem below.

Theorem 1.1 *For each fixed k, there exists a polynomial-time algorithm for the k-*SHORTEST PATHS *restricted to acyclic digraphs.*

We also prove the following inapproximability result, which shows that the problem becomes much harder on general digraphs, even if $k = 2$.

Theorem 1.2 *For each constant c, there is no polynomial-time n^c-approximation algorithm for the* 2-SHORTEST PATHS *unless* P = NP, *where n is the number of vertices of the given digraph.*

With respect to the intractability and inapproximabity of the problem in acyclic digraphs, we show the following theorem.

Theorem 1.3 *For each constant c, there is no polynomial-time n^c-approximation algorithm for the k-*SHORTEST PATHS *restricted to acyclic digraphs unless* P = NP, *where n is the number of vertices of the given digraph.*

Theorems 1.2 and 1.3 show that the result in Theorem 1.1 is tight in the sense that it does not hold, unless P = NP, if we drop either the restriction on k being fixed or on D being acyclic. Surprisingly, the problem becomes much harder if we drop any of these restrictions.

We consider also a variant of the problem in undirected graphs, with multiple pairs of terminals. For this variant, we present a polynomial-time algorithm for the case where all length functions are the same, the given graph is planar and the terminals lie on the boundary of the same face in an adequate order.

2 Disjoint paths in acyclic digraphs

In order to prove Theorem 1.1, we consider the following disjoint paths problem:

$$
\begin{array}{rl}
\textit{given}: & \text{– a directed graph } D = (V, A);\\
& \text{– pairs } (s_1, t_1), \ldots, (s_k, t_k) \text{ of vertices of } D;\\
& \text{– subsets } A_1, \ldots, A_k \text{ of } A;\\
& \text{– a set } H \text{ of pairs } \{i, j\} \text{ from } \{1, \ldots, k\};\\
\textit{find}: & \text{– paths } P_1, \ldots, P_k \text{ in } D \text{ such that:}\\
& \quad (i)\ P_i \text{ is an } s_i\text{-}t_i\text{-path in } D[A_i]\ (i = 1, \ldots, k);\\
& \quad (ii)\ P_i \text{ and } P_j \text{ are vertex-disjoint for } \{i, j\} \text{ in } H.
\end{array}
\tag{1}
$$

Fortune, Hopcroft and Wyllie [3] showed that this disjoint paths problem is NP-hard even for $k = 2$, $A_1 = A_2 = A$ and $H = \{\{1, 2\}\}$. According to Even, Itai and Shamir [2], problem (1) is also NP-hard for acyclic digraphs. In fact, problem (1) is NP-hard even for a fixed acyclic digraph, as noted by Alexander Schrijver. At the end of this section, we include the proof of this unpublished and surprising result.

We prove in the next theorem that problem (1) is polynomially solvable for instances satisfying the following condition:

$$
\begin{array}{l}
\textit{There exists no directed cycle } C = P_{j_0} \cdot P_{j_1} \cdot \ \cdots \ \cdot P_{j_t} \textit{ in } D \textit{ such that:}\\
\quad (i)\ P_{j_i} \textit{ is a path from } u_i \textit{ to } u_{i+1} \textit{ in } D[A_{j_i}],\ u_i \neq t_{j_i}\ (i = 0, \ldots, t),\\
\qquad \textit{where } u_{t+1} = u_0;\\
\quad (ii)\ \{j_0, j_1\}, \ldots, \{j_{t-1}, j_t\}, \{j_t, j_0\} \textit{ belong to } H.
\end{array}
\tag{2}
$$

If P and Q are paths then $P \cdot Q$ denotes the path obtained by the concatenation of P and Q. Note that any acyclic digraph satisfies the condition above. This theorem is a slight generalization of a result by Fortune, Hopcroft and Wyllie [3]. They showed that, for each fixed k, the problem of finding k vertex-disjoint paths in an acyclic digraph is polynomially solvable.

Theorem 2.1 *For each fixed k, there exists a polynomial-time algorithm for the disjoint paths problem (1) for instances satisfying (2).*

Proof. The proof is a minor modification of Schrijver's proof [7] of Fortune, Hopcroft and Wyllie's k vertex-disjoint paths theorem [3, 8]. We include it here for the sake of completeness.

Consider an instance of problem (1), that is, a digraph D, pairs $(s_1, t_1), \ldots, (s_k, t_k)$ of vertices of D, subsets $A_1, \ldots, A_k$ of arcs of D and a set H of pairs $\{i, j\}$ from $\{1, \ldots, k\}$. Make an auxiliary digraph $D' = (V', A')$ as follows. The vertex set V' consists of all k-tuples $(v_1, \ldots, v_k)$ of vertices of D such that $v_i \neq v_j$ for all $\{i, j\}$ in H. There is an arc in D' from $(v_1, \ldots, v_k)$ to $(w_1, \ldots, w_k)$ if and only if there exists an i in $\{1, \ldots, k\}$ such that:

$$
\begin{array}{rl}
(i) & v_j = w_j \text{ for all } j \neq i;\\
(ii) & (v_i, w_i) \text{ is an arc of } A_i;\\
(iii) & \text{if } j \neq i,\ \{i, j\} \in H \text{ and } v_j \neq t_j, \text{ there is no path in } D[A_j] \text{ from } v_j \text{ to } v_i.
\end{array}
\tag{3}
$$

Note that, as k is fixed, the size of D' is polynomially bounded on the size of D. Moreover, the following holds:

$$
\begin{array}{l}
D \text{ contains paths } P_1, \ldots, P_k \text{ such that } P_i \text{ is an } s_i\text{-}t_i\text{-path in } D[A_i]\ (i = 1, \ldots, k)\\
\text{and } P_i \text{ and } P_j \text{ are vertex-disjoint for } \{i, j\} \text{ in } H\\
\qquad\qquad\qquad\qquad \text{if and only if}\\
D' \text{ contains a path } P \text{ from } (s_1, \ldots, s_k) \text{ to } (t_1, \ldots, t_k).
\end{array}
\tag{4}
$$

Suppose that $P_1, \ldots, P_k$ exist. For any i, let P_i follow the vertices $v_{i,0}, v_{i,1}, \ldots, v_{i,t_i}$. So $v_{i,0} = s_i$ and $v_{i,t_i} = t_i$ for each i. Choose $j_1, \ldots, j_k$ such that $0 \leq j_i \leq t_i$ for each i and such that:

(*i*) D' contains a path from $(s_1, \ldots, s_k)$ to $(v_{1,j_1}, \ldots, v_{k,j_k})$, and

(*ii*) $j_1 + \cdots + j_k$ is as large as possible.

Let $I := \{i \mid j_i < t_i\}$. Let us prove by contradiction that $I = \emptyset$. Suppose $I \neq \emptyset$. By the definition of D' and the maximality of $j_1 + \cdots + j_k$, for each i in I, there exists an $i' \neq i$ such that there is a path in $D[A_{i'}]$ from $v_{i',j_{i'}}$ to v_{i,j_i}, with $v_{i',j_{i'}} \neq t_{i'}$ and $\{i', i\}$ in H. So, for i in I, each vertex v_{i,j_i} is an endpoint of a path in $D[A_{i'}]$ starting at another vertex $v_{i',j_{i'}} \neq t_{j_{i'}}$, with i' in I and $\{i, i'\}$ in H. This contradicts (2), so $I = \emptyset$, that is, $j_i = t_i$ for all i, in which case we are done.

Conversely, let P be a path from $(s_1, \ldots, s_k)$ to $(t_1, \ldots, t_k)$ in D'. Let P follow the vertices $(v_{1,j}, \ldots, v_{k,j})$ for $j = 0, \ldots, t$. So $v_{i,0} = s_i$ for $i = 1, \ldots, k$. For each $i = 1, \ldots, k$, let P_i be the path in D following $v_{i,j}$ for $j = 0, \ldots, t$, taking repeated vertices only once. So P_i is an s_i-t_i-path in $D[A_i]$. Moreover, P_i and P_j are vertex-disjoint for each $\{i, j\}$ in H. Indeed, suppose P_1 and P_2 (say) have a vertex in common, where $\{1, 2\}$ belongs to H, that is, $v_{1,j} = v_{2,j'}$ for some $j \neq j'$. Without loss of generality, $j < j'$ and $v_{1,j} \neq v_{1,j+1}$. By the definition of D', there is no path in $D[A_2]$ from $v_{2,j}$ to $v_{1,j}$. This however contradicts the fact that $v_{1,j} = v_{2,j'}$ and that there exists a path in $D[A_2]$ from $v_{2,j}$ to $v_{2,j'}$.

Therefore, to solve problem (1), it is enough to find a path in D' from $(s_1, \ldots, s_k)$ to $(t_1, \ldots, t_k)$, which can be done in polynomial time. ■

The arc-disjoint version of the disjoint paths problem (1) consists of replacing (*ii*) in (1) by:

$$P_i \text{ and } P_j \text{ are arc-disjoint for } \{i, j\} \text{ in } H. \tag{5}$$

This arc-disjoint paths problem can be reformulated in terms of the disjoint paths problem (1). Indeed, let an instance of the arc-disjoint paths problem be given, that is, a digraph $D = (V, A)$, pairs of vertices $(s_1, t_1), \ldots, (s_k, t_k)$, arc sets $A_1, \ldots, A_k$ and a set H of pairs $\{i, j\}$ from $\{1, \ldots, k\}$. We may assume that each s_i is the tail of a unique arc a_i of D and that t_i is the head of a unique arc b_i of D $(i = 1, \ldots, k)$. We make a digraph $D' = (V', A')$ as follows. The vertex set of D' is the arc set A of D (i.e. $V' := A$). There is an arc in D' from a to b if the head of a and the tail of b coincide. For $i = 1, \ldots, k$, we define $A'_i := \{(a, b) \in A' \mid a, b \in A_i\}$. Finally we take $H' := H$.

Finding paths $P_1, \ldots, P_k$ in D satisfying (5) such that P_i is an s_i-t_i-path in $D[A_i]$ $(i = 1, \ldots, k)$ is equivalent to the problem of finding paths $P'_1, \ldots, P'_k$ in D' satisfying (*ii*) of (1) such that P'_i is an a_i-b_i-path in $D'[A'_i]$ $(i = 1, \ldots, k)$. Hence, the arc-disjoint version of problem (1) is polynomially solvable for instances satisfying a condition similar to condition (2).

Now, suppose that, for an instance of problem (1), one is given also non-negative functions $l_1, \ldots, l_k$ on the arcs of D. Then it is possible to find in polynomial time a solution $P_1, \ldots, P_k$ of problem (1) such that $\sum_{i=1}^{k} l_i(P_i)$ is as small as possible. Just define a length function on the arcs of D' (the digraph from the proof of Theorem 2.1) as follows. The length of an arc of D' from $(v_1, \ldots, v_k)$ to $(w_1, \ldots, w_k)$ satisfying (3) is $l_i(v_i, w_i)$. Now, a shortest path from $(s_1, \ldots, s_k)$ to $(t_1, \ldots, t_k)$ in D' with this length function on its arcs gives the desired paths. As the shortest path problem in an acyclic digraph with arbitrary length on its arcs can be solved in linear time, we have Theorem 1.1.

Theorem 1.1 *For each fixed k, there exists a polynomial-time algorithm for the k-*SHORTEST PATHS *restricted to acyclic digraphs.* ■

We conclude this section with the proof of Schrijver's result on the complexity of problem (1).

Theorem 2.2 (A. Schrijver) *The disjoint paths problem (1) restricted to instances having the digraph in Figure 1 as input is* NP*-hard.* ■

Proof. Consider the following transformation of PLANAR 3-COLORABILITY to problem (1) restricted to instances having the acyclic digraph displayed in Figure 1 as input. A *k-coloring* of a graph $G = (V, E)$ is a function f from V to $\{1, \ldots, k\}$ such that $f(u) \neq f(v)$ whenever $\{u, v\}$ belongs to E. Graph G is *k-colorable* if G has a k-coloring. The PLANAR 3-COLORABILITY problem consists of: *given:* a planar graph $G = (V, E)$; *question:* is G 3-colorable? PLANAR 3-COLORABILITY was shown to be NP-complete by Stockmeyer [9].

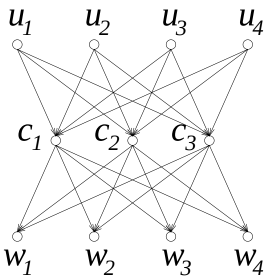

Figure 1: Problem (1) is NP-hard for instances having this acyclic digraph as input.

Let us be given a planar graph $G = (V, E)$ with $V = \{v_1, \ldots, v_k\}$ and let f' be a 4-coloring of G. This function f' can be computed in polynomial time (see, for instance, Nishizeki and Chiba [5]). We construct an instance of problem (1) depending on G and f' as follows. Let $H := \{\{i, j\} \mid \{v_i, v_j\} \in E\}$ and, for $i = 1, \ldots, k$, let $s_i := u_{f'(v_i)}$, $t_i := w_{f'(v_i)}$ and A_i be the set of all arcs of the digraph in Figure 1. We claim that G is 3-colorable if and only if the constructed instance of problem (1) is feasible. Suppose G is 3-colorable and let f be a 3-coloring of G. For $i = 1, \ldots, k$, let P_i be the path from s_i to t_i that traverses $c_{f(v_i)}$. One can check that $P_1, \ldots, P_k$ is a solution to the problem (1). Conversely, let $P_1, \ldots, P_k$ be a solution to the disjoint paths problem (1) and define f from V to $\{1, 2, 3\}$ such that, for $i = 1, \ldots, k$, the path P_i from s_i to t_i traverses the vertex $c_{f(v_i)}$. One can verify that f is a 3-coloring of G. ■

3 Inapproximability for the 2-SHORTEST PATHS

In this section we analyze the complexity of the 2-SHORTEST PATHS problem. Specifically, we prove Theorem 1.2.

Theorem 1.2 *For each constant c, there is no polynomial-time n^c-approximation algorithm for the* 2-SHORTEST PATHS *unless* P = NP, *where n is the number of vertices of the given digraph.*

Proof. We may assume $c \geq 1$. Suppose that there is a polynomial-time n^c-approximation algorithm A for the 2-SHORTEST PATHS, where n is the number of vertices of the given digraph. Let us show that, if this is the case, we can solve 3-SAT in polynomial time, which implies that P = NP. For this, consider the following polynomial-time reduction from 3-SAT to 2-SHORTEST PATHS.

Let Φ be an instance of 3-SAT, that is, a set $\{C_1, \ldots, C_m\}$ of 3-clauses on variables $x_1, \ldots, x_h$. Let us describe a digraph D, two length functions l_1 and l_2 on the arcs of D and two vertices s and t.

For each variable x_i, denote by d_i the largest between the number of times x_i appears in Φ and the number of times that $\overline{x}_i$ appears in Φ. There is a gadget as in Figure 2(a) for each x_i. The number of undirected four-cycles in the gadget is $d_i + 1$. The source vertex in the gadget is called v_i and the sink vertex, w_i. The vertices of in-degree one in the gadget are partitioned into two sets: L_i and R_i, as in Figure 2(a).

For each clause C_j, there is a gadget as in Figure 2(b). The sink and source vertices are called u_j and l_j respectively. Each of the other vertices has as label one of the literals in clause C_j.

The digraph of the instance of 2-SHORTEST PATHS is obtained as follows. First, we connect the gadgets of all variables and clauses in series, identifying w_i and v_{i+1} ($i = 1, \ldots, n-1$) and u_j and l_{j+1} ($j = 1, \ldots, m-1$). Then, we add an arc from s to v_1, one from w_h to l_1 and one from u_m to t. The arcs we have up to now

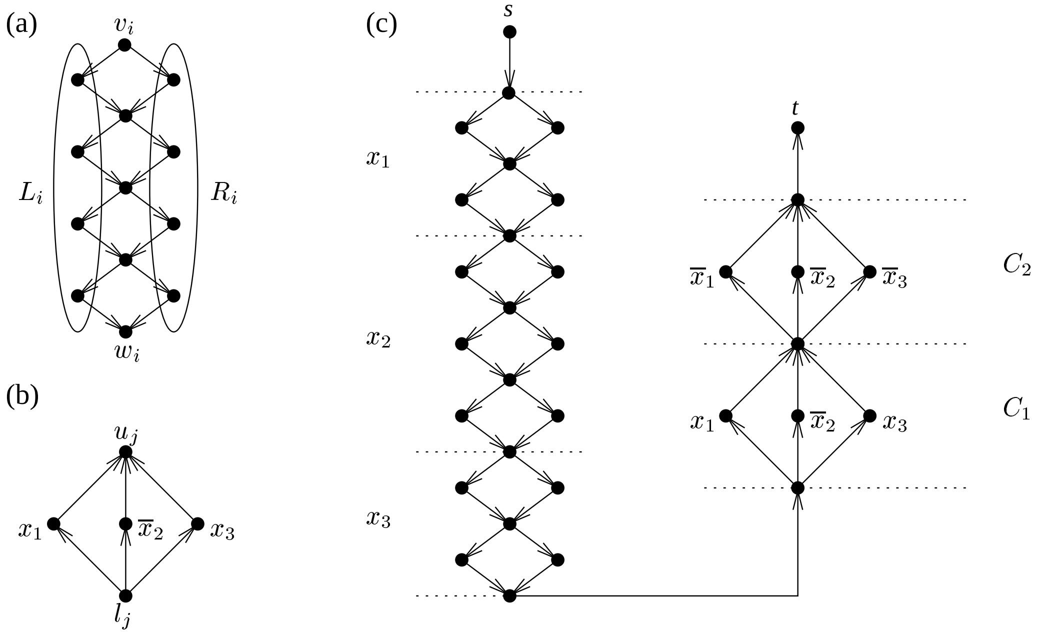

Figure 2: (a) The gadget for variable x_i. One, between x_i or $\overline{x}_i$, appears three times in Φ, while the other appears at most three times. (b) The gadget for clause $C_j = \{x_1, \overline{x}_2, x_3\}$. (c) Arcs of type 1 of the digraph built from $\Phi = (x_1 \vee \overline{x}_2 \vee x_3)(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3)$.

are said to be of type 1. See Figure 2(c). Second, we add three arcs from s: one to t, one to the first vertex in L_1 and another to the first vertex in R_1. Similarly, we add two arcs to t: one from the last vertex in L_h and one from the last vertex in R_h. For each two consecutive vertices in L_i, we add a path from the upper one to the lower one, of length one or two. When the path has length two, the middle vertex is one of the vertices labeled x_i in the clause gadgets. The same holds for R_i with $\overline{x}_i$ in the place of x_i. This is done in such a way that any labeled vertex is in exactly one of these two-length paths. Finally, there are also arcs from the last vertex in L_i and from the last vertex in R_i to both, the first vertex in L_{i+1} and the first vertex in R_{i+1} ($i = 1, \ldots, n-1$). The arcs added in this second phase are said to be of type 2. This finishes the description of the digraph D and vertices s and t. See Figure 3(a) for a complete example. Note that the number of vertices in this digraph is at most $4 + 3d + 3h + 4m$, where $d := \sum_{i=1}^{h} d_i \leq 3m$. Also, there are two internally disjoint paths from s to t in D.

To complete the description of the instance of 2-SHORTEST PATHS, it is missing only to describe the two length functions l_1 and l_2 on the arcs of D. In l_1, arcs of type 1 have length one, while arcs of type 2 have length $M := (4 + 3d + 3h + 4m)^{c+1} + 1$. In l_2, all arcs have length one, but arc st, whose length is M.

Note that the construction of D, s, t, l_1 and l_2 takes polynomial time on the size of Φ.

Claim 3.1 *Φ is satisfiable if and only if there are two internally disjoint paths P_1 and P_2 from s to t in D such that $l_1(P_1) + l_2(P_2) \leq 4 + 3d + 3h + 4m$.*

Proof. Assume Φ is satisfiable and consider an assignment which satisfies Φ. Let us describe two internally disjoint paths P_1 and P_2 in D from s to t such that $l_1(P_1) + l_2(P_2) \leq 4 + 3d + 3h + 4m$.

Path P_1 starts with arc sv_1, goes from v_1 to w_h using only arcs in the variable gadgets, then uses arc $w_h l_1$ and goes from l_1 to u_m using only arcs in the clause gadgets. It ends with arc $u_m t$. Inside the variable gadget for x_i, path P_1 goes through all vertices in L_i if x_i is TRUE in the assignment or all vertices in R_i if x_i is FALSE. In the clause gadgets, P_1 goes always through a vertex whose label is a TRUE literal in the assignment. Note that P_1 uses only type 1 arcs. Thus $l_1(P_1) = 3 + 2d + 2h + 2m$.

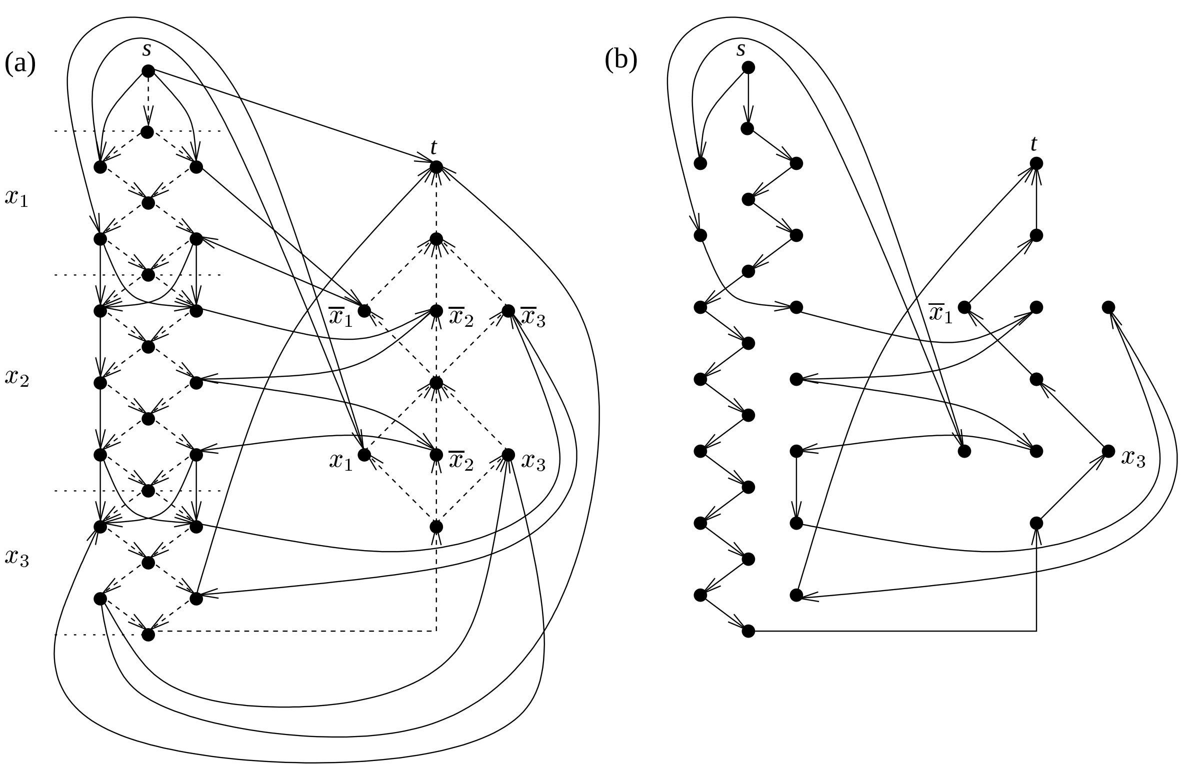

Figure 3: (a) Digraph built from $\Phi = (x_1 \vee \overline{x}_2 \vee x_3)(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3)$. The dashed arcs have l_1 equals one, while the others have l_1 equals M. (b) Paths P_1 and P_2 corresponding to the assignment $x_1 = F$, $x_2 = T$ and $x_3 = T$.

Path P_2 uses only type 2 arcs. If x_1 is TRUE (FALSE), it goes from s to the first vertex in L_1 (R_1). From there, it traverses all vertices in L_1 (R_1) and jumps to L_2 if x_2 is TRUE or to R_2 if x_2 is FALSE and proceeds in the same way until it gets to the last vertex in R_h or L_h. In this traversal, it goes back and forth to the clause gadgets through some length-two paths, always using a vertex whose label is a literal set to FALSE. From the last vertex in L_h or R_h, it goes directly to t. Note that P_2 is indeed internally disjoint from P_1, as it uses only type 2 arcs. Moreover, $l_2(P_2) = 1 + d + h + 2m$.

Therefore $l_1(P_1) + l_2(P_2) = 4 + 3d + 3h + 4m$, as desired. See in Figure 3(b) how P_1 and P_2 look like for the example given in Figure 3(a).

Now assume there are two internally disjoint paths P_1 and P_2 in D from s to t such that $l_1(P_1)+l_2(P_2) \leq 4 + 3d + 3h + 4m$. Note that P_1 can only use type 1 arcs, otherwise $l_1(P_1) \geq M > 4 + 3d + 3h + 4m$ (the last inequality holds as $c \geq 1$). Also, P_2 does not use arc st, as $l_2(st) = M > 4 + 3d + 3h + 4m$. As P_1 uses only type 1 arcs, P_1 uses sv_1, then it goes from v_1 to w_h using only arcs in the variable gadgets, then it uses $w_h l_1$ and goes from l_1 to u_m inside the clause gadgets, finishing with $u_m t$. Path P_1 cannot use vertices both in L_i and R_i, otherwise the only path from s to t in D internally disjoint from P_1 consists of st. But $l_2(st) = M > 4 + 3d + 3h + 4m$. Indeed, P_1 must pass by all vertices of the variable gadget x_i not in $L_i \cup R_i$ and by all unlabeled vertices in the clause gadgets. But then, if P_2 uses the first vertex in L_i, it has no other way except using all other vertices in L_i. The same holds for R_i.

Now we are ready to describe the assignment. Set x_i to TRUE if and only if P_2 uses vertices of L_i. Note that P_2 visits all labeled vertices in the clause gadgets whose labels were set to FALSE. But path P_1 necessarily uses a labeled vertex in each clause gadget. The label $\tilde{x}_i$ of this labeled vertex must then be TRUE, which means there is a TRUE literal in each clause. That is, Φ is satisfiable. ■

Now we proceed with the proof of Theorem 1.2. Run algorithm A on the constructed instance of 2-SHORTEST PATHS. The algorithm returns two paths, P_1 and P_2. If $l_1(P_1)+l_2(P_2) < M$ then Φ is satisfiable, otherwise Φ is not satisfiable.

First, note that the above algorithm runs in polynomial-time, as the reduction and A take polynomial-time. Moreover, it solves 3-SAT. Indeed, assume $l_1(P_1) + l_2(P_2) \geq M$. As A is an n^c-approximation and $n = 4 + 3d + 3h + 4m$, the value of an optimal solution for this instance of the 2-SHORTEST PATHS is at least $M/(4 + 3d + 3h + 4m)^c > 4 + 3d + 3h + 4m$. By the claim, Φ is not satisfiable. Now, assume $l_1(P_1) + l_2(P_2) < M$. This means P_1 uses no type 2 arc. Any path from s to t in D which uses no type 2 arc uses exactly $3 + 2d + 2h + 2m$ arcs of type 1, that is, $l_1(P_1) = 3 + 2d + 2h + 2m$. But then, as $n = 4 + 3d + 3h + 4m$, path P_2 uses at most $2 + d + h + 2m$ vertices, that is, $l_2(P_2) \leq 2 + d + h + 2m$. Therefore, by the claim, Φ is satisfiable. As 3-SAT is solvable in polynomial time only if P = NP, there is no n^c-approximation algorithm for 2-SHORTEST PATHS unless P = NP. ■

In fact, it is easy to modify this theorem to show that, for any polynomial-time computable function f, there is no polynomial-time $f(|I|)$-approximation algorithm for 2-SHORTEST PATHS, where I denotes an arbitrary instance of 2-SHORTEST PATHS.

Consider the undirected edge/vertex-disjoint versions of the k-SHORTEST PATHS problem. There are well-known reductions from the undirected edge-disjoint version to the undirected vertex-disjoint and to the directed arc/vertex-disjoint versions of the problem. One can modify the proof of the previous theorem in order to get the following stronger theorem.

Theorem 3.2 *For each constant c, there is no polynomial-time n^c-approximation algorithm for the undirected edge-disjoint* 2-SHORTEST PATHS *unless* P = NP, *where n is the number of vertices of the given graph.*

4 Inapproximability for acyclic digraphs

In this section we show that if k is non-fixed then the k-SHORTEST PATHS is hard to approximate, even restricted to acyclic digraphs. The proof of the theorem below is a modification of the proof of Theorem 1.2.

Theorem 1.3 *For each constant c, there is no polynomial-time n^c-approximation algorithm for the k-*SHORTEST PATHS *restricted to acyclic digraphs unless* P = NP, *where n is the number of vertices of the given digraph.*

Proof. We may assume $c \geq 1$. Suppose that there is a polynomial-time n^c-approximation algorithm A for the k-SHORTEST PATHS on acyclic digraph, where n is the number of vertices of the given digraph. Consider the following polynomial-time reduction from 3-SAT to k-SHORTEST PATHS.

Let Φ be an instance of 3-SAT, that is, a set $\{C_1, \ldots, C_m\}$ of 3-clauses on variables $x_1, \ldots, x_h$. Let us describe an acyclic digraph D, two vertices s and t and length functions $l_0, \ldots, l_h$ on the arcs of D.

Digraph D consists of vertices $s, v_0, \ldots, v_h, t$, arcs st, sv_0, v_ht and, for each variable x_i, two internally disjoint paths, Q_i and $\bar{Q}_i$, from v_{i-1} to v_i. Path Q_i ($\bar{Q}_i$) has as many internal vertices as appearances of x_i ($\overline{x}_i$). Additionally, for each clause C_j, there are three length-two paths from s to t, each one having as middle vertex a vertex labeled by one of the literals in C_j. This is done in such a way that no two of these paths share the middle vertex. See Figure 4 for an example.

Let $M := (3 + 3m + h)^c(2 + 3m + 3h) + 1$. For each C_j, set $l_j(e) := 1$ if e is one of the arcs in the three length-two paths added for C_j and set $l_j(e) := M$ otherwise. Set $l_0(e) := 1$ if $e = sv_0$ or $e = v_ht$ or e is in path Q_i or $\bar{Q}_i$ for some i. Otherwise set $l_0(e) := M$. This completes the description of the instance of k-SHORTEST PATHS. Observe that D is acyclic and that there are k internally disjoint paths from s to t in D. Also, observe that D, s, t and $l_0, \ldots, l_h$ can be constructed in polynomial time.

Claim 4.1 *Φ is satisfiable if and only if there are internally disjoint paths $P_0, \ldots, P_h$ from s to t in D*

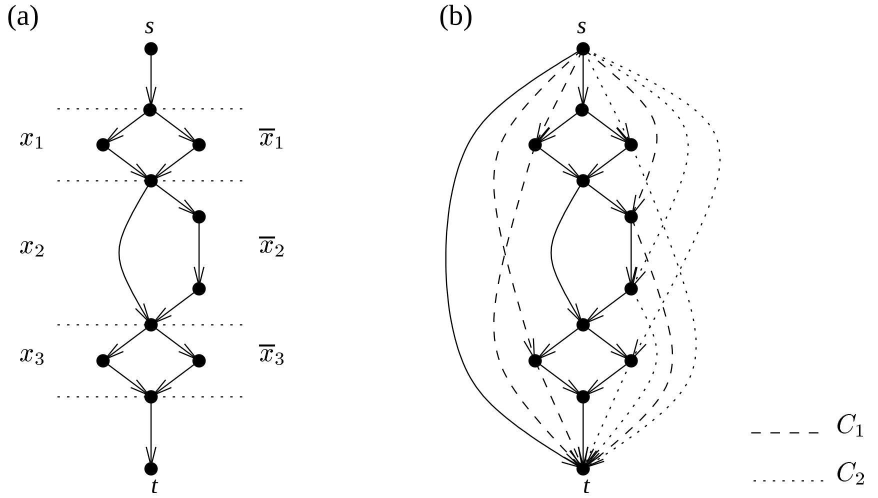

Figure 4: (a) Arcs for the variables. (b) Digraph for $\Phi = (x_1 \vee \overline{x}_2 \vee x_3)(\overline{x}_1 \vee \overline{x}_2 \vee \overline{x}_3)$.

such that $l_0(P_0) + \cdots + l_h(P_h) \leq 2 + 3m + 3h$.

Proof. Assume Φ is satisfiable and consider an assignment which satisfies Φ. Let P_0 be the path starting with arc sv_0, ending with arc $v_h t$, and using Q_i, if x_i is FALSE, or $\bar{Q}_i$, if x_i is TRUE, for each i. Note that each labeled vertex in P_0 has as label a literal that is FALSE in the assignment. Moreover, $l_0(P_0) \leq 2 + 3m + h$. For each clause C_j, let P_j be one among the three paths for C_j that use a vertex whose label is a literal set to TRUE in the assignment. There is one such path because the assignment satisfies Φ. Paths $P_0, \ldots, P_h$ are such that $l_0(P_0) + \cdots + l_h(P_h) \leq 2 + 3m + 3h$.

Suppose now that there are internally disjoint paths $P_0, \ldots, P_h$ from s to t in D such that $l_0(P_0) + \cdots + l_h(P_h) \leq 2 + 3m + 3h$. Note that $M > 2 + 3m + 3h$, because $c \geq 1$. Therefore, $P_0, \ldots, P_h$ use only arcs whose length is one in their respective length functions. In particular, P_0 uses necessarily arcs sv_0 and $v_h t$ and passes by vertices $v_1, \ldots, v_{h-1}$. To go from v_{i-1} to v_i, path P_0 uses either Q_i or $\bar{Q}_i$. Set variable x_i to TRUE if P_0 uses $\bar{Q}_i$ and set x_i to FALSE if P_0 uses path Q_i. For each C_j, path P_j has to be one of the length-two paths for C_j. Let $\tilde{x}_i$ be the label of the middle in P_j. If P_0 uses Q_i, then $\tilde{x}_i = \overline{x}_i$ (or P_j and P_0 would not be internally disjoint). If P_0 uses $\bar{Q}_i$, then $\tilde{x}_i = x_i$. In both cases, $\tilde{x}_i$ is TRUE in the assignment and therefore this assignment satisfies C_j, for all j. ■

To complete the proof of Theorem 1.3, it is enough to describe how to use algorithm A to get a polynomial-time algorithm for 3-SAT. Just run algorithm A on the constructed instance of k-SHORTEST PATHS. It returns paths $P_0, \ldots, P_h$. If $l_0(P_0) + \cdots + l_h(P_h) < M$, then Φ is satisfiable, otherwise Φ is not satisfiable. The resulting algorithm is clearly polynomial and solves 3-SAT. Indeed, assume that $l_0(P_0) + \cdots + l_h(P_h) \geq M$. Algorithm A is an n^c-approximation, where n is the number of vertices of D, that is, $n = 3 + 3m + h$. Therefore, the value of an optimal solution for this instance of the k-SHORTEST PATHS is at least $M/(3 + 3m + h)^c > 2 + 3m + 3h$. By the claim, Φ is not satisfiable. Now, assume $l_0(P_0) + \cdots + l_h(P_h) < M$. This means P_0 uses neither st nor arcs in the paths of the clauses. Therefore P_0 uses arcs sv_0 and $v_h t$ and, for each i, uses either Q_i or $\bar{Q}_i$. Thus $l_0(P_0) \leq 2 + 3m + h$. Also, each path P_j has to be one of the paths for clause C_j, otherwise $l_j(P_j) \geq M$. Hence $l_j(P_j) = 2$, for all j, and $l_0(P_0) + \cdots + l_h(P_h) \leq 2 + 3m + 3h$. By the claim, Φ is satisfiable. ■

This theorem also holds for any polynomial-time computable function f: there is no polynomial-time $f(|I|)$-approximation algorithm for k-SHORTEST PATHS in acyclic digraphs. Here, I denotes an arbitrary instance of k-SHORTEST PATHS in acyclic digraphs.

5 Minimizing sum of lengths

Consider the following shortest disjoint paths problem:

given : – an undirected planar graph $G = (V, E)$, embedded in $\mathbb{R}^2$;
– pairs $\{s_1, t_1\}, \ldots, \{s_k, t_k\}$ of vertices on the boundary of G;
– a non-negative function l on the edges of G;

find : – pairwise vertex-disjoint paths $P_1, \ldots, P_k$ in G where P_i is an s_i-t_i-path, for each $i = 1, \ldots, k$, and $l(P_1) + \cdots + l(P_k)$ is as small as possible.

We denote this problem by $\text{SDP}(G, \{s_1, t_1\}, \ldots, \{s_k, t_k\})$, or simply by SDP, when the instance is clear from the context.

If the vertices $s_1, \ldots, s_k, t_k, \ldots, t_1$ occur in this order when following the boundary of G, then SDP can be seen as a particular case of the min-cost flow problem. Indeed, from G, we construct a digraph D by splitting each vertex v of G into two vertices v^+ and v^-, joined by an arc (v^-, v^+) of cost zero. An edge vw of G becomes arcs (v^+, w^-) and (w^+, v^-) of D, both of cost $l(vw)$. In addition, D has vertices s, t and arcs (s, s_i^-), (t_i^+, t), for $i = 1, \ldots, k$, of cost zero. Solving SDP is equivalent to finding a maximum s-t-flow of minimum cost in D, with each arc having capacity one. Hence, in this particular case, SDP can be solved in polynomial time.

A graph $G = (V \cup \{c\}, E)$ is called a *wheel* if $G - c$ is a circuit and $\{v, c\} \in E$ for each v in V. Grötschel, Martin and Weismantel [4] showed that if $G = (V \cup \{c\}, E)$ is a wheel and $s_1, t_1, \ldots, s_k, t_k$ occur in this order when following the circuit $G - c$ then the edge-disjoint version of SDP can be solved in polynomial time. Moreover, Grötschel, Martin and Weismantel gave a complete description of the *path packing polytope* (the convex hull of incident vectors of sets $E' \subseteq E$, such that $G[E']$ is a packing of edge-disjoint s_i-t_i-paths in G).

Using dynamic programming, we show that, also in the following case, SDP can be solved in polynomial time:

$$\begin{array}{l} k \text{ is fixed and the vertices } s_1, t_1, \ldots, s_k, t_k \text{ occur in this} \\ \text{order when following the boundary of } G. \end{array} \tag{6}$$

We assume that SDP is feasible (this can be tested in linear time) and that G is 2-connected.

We shall use the following notation. If P is a path and u, v are vertices in P then $P(u, v)$ is the subpath of P connecting u to v. We denote by Q_i a shortest s_i-t_i-path. Let R_i be the subgraph of G induced by the vertices in the closed region bounded by the path Q_i and the path on the boundary of G from s_i to t_i containing no other s_j or t_j ($i = 1, \ldots, k$). Also, let $Q_{i,j} := R_i \cap R_j$. By shortcut arguments, we may assume that $Q_{i,j}$ is either a path or empty, for $i \neq j$. Figure 5 illustrates the notation. There, P_1, P_2, P_3 are paths of an optimal solution.

Let G, $\{s_1, t_1\}, \ldots, \{s_k, t_k\}$ and l be an instance of SDP with the property that $s_1, t_1, \ldots, s_k, t_k$ occur in this order when following the boundary of G. Observe that SDP has an optimal solution $P_1, \ldots, P_k$ so that P_i is entirely contained in the region R_i ($i = 1, \ldots, k$).

Consider some i and j with $i \neq j$. Let $Q_{i,j} := (v_0, e_1, v_1, \ldots, e_d, v_d)$. We call (f, h) a *possible choice* (for (i, j)) if $f = h = \mathsf{nil}$ or $f = v_p$ and $h = v_q$ for some p, q satisfying $0 \leq p \leq q \leq d$. We say that (f, h, f', h') is a *feasible choice* (for (i, j)) if (f, h) and (f', h') are possible choices and if $f \neq \mathsf{nil} \neq f'$ implies $\{f, h\} \cap \{f', h'\} = \emptyset$. For each feasible choice (f, h, f', h'), let

$$\begin{aligned} G_{i,j,f,h,f',h'} \quad := \quad & G[V(R_i) \setminus (V(Q_{i,j}) \setminus V(Q_{i,j}(f, h)))] \\ & \cup G[V(R_j) \setminus (V(Q_{i,j}) \setminus V(Q_{i,j}(f', h')))], \end{aligned}$$

where $Q_{i,j}(\mathsf{nil},\mathsf{nil}) := \emptyset$. Finally, we say that a sequence

$$((f_{i,j}, h_{i,j}, f'_{i,j}, h'_{i,j}) : 1 \leq i < j \leq k)$$

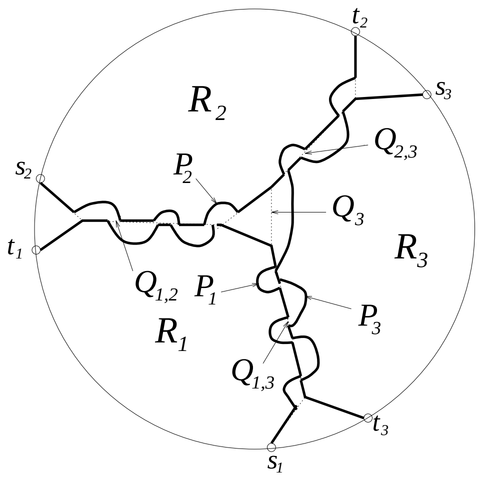

Figure 5: Illustration of the definitions.

is an *acceptable choice* provided that $(f_{i,j}, h_{i,j}, f'_{i,j}, h'_{i,j})$ is a feasible choice for each (i,j), for all $1 \leq i < j \leq k$.

Our dynamic programming approach is based on the following optimality criterion:

> Let $P_1, \ldots, P_k$ be an optimal solution to SDP and let f_i, h_i, f_j, h_j be the first and last vertices of $Q_{i,j}$ in P_i and in P_j, respectively, for some i, j with $1 \leq i < j \leq k$. Then $P_i(f_i, h_i), P_j(f_j, h_j)$ is an optimal solution to $\textsc{sdp}(G_{i,j,f_i,h_i,f_j,h_j}, \{f_i, h_i\}, \{f_j, h_j\})$ (see Figure 6).

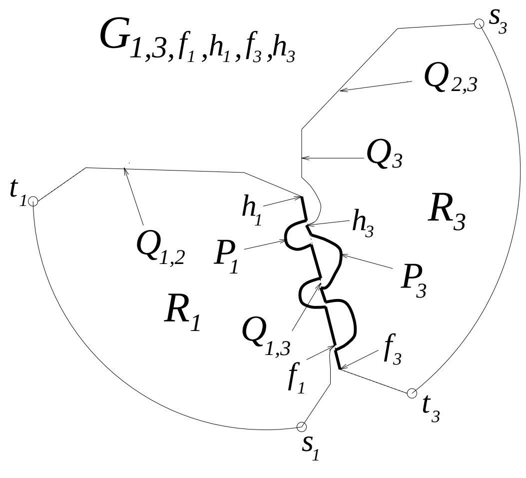

Figure 6: Illustration of the optimality condition for SDP.

The algorithm consists of first computing, for each (i,j), with $1 \leq i < j \leq k$, and for each feasible choice (f, h, f', h') for (i,j), a solution $P_{i,j,f,h,f',h'}$, $P_{j,i,f,h,f',h'}$ to $\textsc{sdp}(G_{i,j,f,h,f',h'}, \{f, h\}, \{f', h'\})$, where, $P_{i,j,\mathsf{nil},\mathsf{nil},f',h'} := \emptyset$ and $P_{j,i,f,h,\mathsf{nil},\mathsf{nil}} := \emptyset$). Now, we enumerate all acceptable choices

$$A \quad := \quad ((f_{i,j}, h_{i,j}, f'_{i,j}, h'_{i,j}) : i, j = 1, \ldots, k, i < j)$$

(so, $(f_{i,j}, h_{i,j}, f'_{i,j}, h'_{i,j})$ is a feasible choice for all (i,j) with $1 \leq i < j \leq k$) and we compute $P_1^A, \ldots, P_k^A$, where P_i^A is a shortest s_i-t_i-path in

$$G[(V(R_i) \setminus V(Q_{i,j})) \;\cup\; (\cup_{i \neq j} V(P_{i,j,f_{i,j},h_{i,j},f'_{i,j},h'_{i,j}}))] \quad (i = 1, \ldots, k),$$

if there exists any.

The algorithm returns, for some acceptable choice A^*, vertex-disjoint paths $P_1^{A^*}, \ldots, P_k^{A^*}$ so that,

$$\sum_{i=1}^{k} l(P_i^{A^*}) = \min\{\sum_{i=1}^{k} l(P_i^{A}) \mid A \text{ is an acceptable choice}\}.$$

Theorem 5.1 *If $s_1, t_1, \ldots, s_k, t_k$ occur in this order when following the boundary of G then, for each fixed k,* SDP *can be solved in polynomial time.*

Proof. Any solution $P_1, \ldots, P_k$ to SDP such that P_i is a subgraph of R_i induces an acceptable choice to SDP. (For each (i, j) with $i < j$, we define $(f_{i,j}, h_{i,j}, f'_{i,j}, h'_{i,j})$ as the first and last vertices of $Q_{i,j}$ in P_i and P_j, respectively. If the path $Q_{i,j}$ does not meet, say, P_i then $f_{i,j} := h_{i,j} := \mathsf{nil}$.) Since by shortcut arguments there exists at least one such a solution, it follows that the algorithm generates and returns a solution to SDP.

One sees that $\text{SDP}(G_{i,j,f,h,f',h'}, \{f, h\}, \{f', h'\})$ can be solved in polynomial time, as (if $f \neq \mathsf{nil} \neq f'$) f, h, f', h' are vertices on the boundary of $G_{i,j,f,h,f',h'}$. Thus, in order to show polynomiality, it remains to verify that the number of acceptable choices is polynomially bounded. Indeed,

$$|\{Q_{i,j} \mid i < j \text{ and } Q_{i,j} \neq \emptyset\}| = O(k^2)$$

and there exist $O(n^4)$ feasible choices for each (i, j). Hence, there are $O(n^{4k^2})$ acceptable choices. ■

Remark. Using similar techniques one can prove that if $s_1, t_1, \ldots, s_k, t_k$ occur in this order when following the boundary of G then the edge-disjoint version of SDP is polynomially solvable for each fixed k. It can also be proved that if, in addition, $|i - j| > 1$ implies $Q_{i,j} = \emptyset$, then SDP can be solved in polynomial time (here k does not need to be fixed).

Acknowledgments

We thank Bruce Shepherd for bringing the k-SHORTEST PATHS problem to our attention. We also thank Alexander Schrijver and Bruce Shepherd for many valuable comments and suggestions.

References

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network flows*, Prentice-Hall, Englewood Cliffs, NJ, 1993.

[2] S. Even, A. Itai, and A. Shamir, *On the complexity of timetable and multicommodity flow problems*, SIAM Journal on Computing **5** (1976), 691–703.

[3] S. Fortune, J. Hopcroft, and J. Wyllie, *The direct subgraph homeomorphism problem*, Theoretical Computer Science **10** (1980), 111–121.

[4] M. Grötschel, A. Martin, and R. Weismantel, *Optimum path packing on wheels: the consecutive case*, Comput. Math. Appl. **31** (1996), no. 11, 23–35.

[5] T. Nishizeki and N. Chiba, *Planar graphs: Theory and algorithms*, North-Holland, Amsterdam, 1988.

[6] Y. Perl and Y. Shiloach, *Finding two disjoint paths between two pairs of vertices in a graph*, Journal of the Association for Computing Machinery **25** (1978), 1–9.

[7] A. Schrijver, *A group-theoretical approach to disjoint paths in directed graphs*, CWI Quarterly **6** (1993), 257–266.

[8] ———, *Combinatorial optimization—polyhedra and efficiency*, Springer-Verlag, forthcoming book, Part VII. Multiflows and Disjoint Paths.

[9] L.J. Stockmeyer, *Planar 3-colorability is* NP*-complete*, SIGACT News **5** (1973), 19–25.