Length-Bounded Disjoint Paths in Planar Graphs

H. van der Holst
Centrum voor Wiskunde en Informatica
P.O. Box 94079, 1090 GB Amsterdam
The Netherlands
E-mail address: `hein@cwi.nl`

J.C. de Pina*
Instituto de Matemática e Estatística, Universidade de São Paulo
Rua do Matão 1010, 05508–900 São Paulo
Brazil
E-mail address: `coelho@ime.usp.br`

Abstract

The following problem is considered: *given*: an undirected planar graph $G = (V, E)$ embedded in $\mathbb{R}^2$, distinct pairs of vertices $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ of G adjacent to the unbounded face, positive integers $b_1, \ldots, b_k$ and a function $l : E \to \mathbb{Z}_+$; *find*: pairwise vertex-disjoint paths $P_1, \ldots, P_k$ such that for each $i = 1, \ldots, k$, P_i is a r_i-s_i-path and the sum of the l-length of all edges in P_i is at most b_i It is shown that the problem is NP-hard in the strong sense. A pseudo-polynomial time algorithm is given for the case of $k = 2$.

1991 Mathematical Subject Classification: 05C38, 05C85, 68Q25, 90C39.
Key words and phrases: planar graph, disjoint paths, complexity, dynamic programming.

1 Introduction

Most of the research on disjoint paths problems has concentrated on whether or not a given graph has a collection of disjoint paths between specific terminals. In the present paper we apply dynamic programming techniques in order to find a collection of "short" disjoint paths between specific endpoints in a planar graph. These problems turns out to be much harder than just finding disjoint paths; even simple cases where finding disjoint paths is easy are NP-hard for finding short disjoint paths. More specifically, we consider the following *length-bounded disjoint paths problem*:

$$\begin{array}{rl} \textit{given}: & \text{– an undirected planar graph } G = (V, E) \text{ embedded in } \mathbb{R}^2; \\ & \text{– pairs } \{r_1, s_1\}, \ldots, \{r_k, s_k\} \text{ of vertices of } G; \\ & \text{– positive integers } b_1, \ldots, b_k; \\ & \text{– a `length' function } l : E \to \mathbb{Z}_+; \\ \textit{find}: & \text{– pairwise vertex-disjoint paths } P_1, \ldots, P_k \text{ such that for each} \\ & \quad i = 1, \ldots, k,\ P_i \text{ is an } r_i\text{-}s_i\text{-path and } \mathrm{length}_l(P_i) \le b_i. \end{array} \tag{1}$$

*Part of this work was done while this author was a visitor at the Centrum voor Wiskunde en Informatica on leave from University of São Paulo. Partially supported by FAPESP (Proc. 90/1173-2, 96/04505–2), and by MCT/FINEP (PRONEX project 107/97).

(A *path* is a sequence $(v_0, e_1, v_1, \ldots, e_t, v_t)$ of vertices and edges such that e_j connects v_{j-1} and v_j $(j = 1, \ldots, t)$. If $P_i = (r_i = v_0, e_1, v_1, \ldots, e_t, v_t = s_i)$ then $\text{length}_l(P_i) = \sum_{i=1}^{t} l(e_i)$.)
This problem also seems to be considerably harder than the following similar *shortest disjoint paths problem*: *given*: an undirected planar graph $G = (V, E)$ embedded in $\mathbb{R}^2$, distinct pairs of vertices $\{r_1, s_1\}, \ldots, \{r_k, s_k\}$ of G adjacent to the unbounded face, a positive integer b and a function $l : E \to \mathbb{Z}_+$; *find*: pairwise vertex-disjoint paths $P_1, \ldots, P_k$ such that for each $i = 1, \ldots, k$, P_i is a r_i-s_i-path and $\sum_{i=1}^{k} \text{length}_l(P_i) \leq b$. While the shortest disjoint paths problem is polynomially solvable for non-fixed k and $r_1, r_2, \ldots, r_k, s_k, \ldots, s_2, s_1$ occurring in order when following the boundary of G (say) clockwise (via a reduction to the min-cost flow problem), problem (1) is easily proved to be NP-hard for $k = 2$ (see Section 2).

Li, McCormick and Simchi-Levi [6] showed that if the planarity condition is dropped then problem (1) is NP-hard in the strong sense for $k = 2$. Moreover, extending techniques in Perl and Shiloach [9], a pseudo-polynomial time algorithm is describe in [6] which given a acyclic directed graph D and distinct vertices r and s of D finds k (fixed) length-bounded internally vertex-disjoint paths from r to s.

Itai, Perl and Shiloach [4] considered the related problem of finding in an undirected graph the maximum number of internally vertex-disjoint paths of length $L \in \mathbb{Z}_+$ and of length bounded by $L \in \mathbb{Z}_+$ connecting two given vertices r and s. They showed that for $L \geq 4$ ($L \geq 5$ for the bounded case) these problems are all NP-hard. In this paper we prove that problem (1) is NP-hard in the strong sense.

Applying dynamic programming techniques, similar to the ones used by [6, 9], we show the pseudo-polynomial time solvability of the length-bounded disjoint path problem (1) for $k = 2$.

The technique we apply here consists of reformulating these disjoint paths problems in terms of finding a shortest path in a certain (large) acyclic digraph. The *integer knapsack problem* is a well-known example of a combinatorial optimization problem that can be reformulated in these terms. In particular, from this reformulation one derives a pseudo-polynomial time algorithm for the integer knapsack problem (see, e.g., Chapter 16 of [8]). Actually, integer linear programming itself can be viewed as a shortest path problem in a large acyclic graph (cf. [11]), this yields a pseudo-polynomial time algorithm for integer linear programming if the number of constraints is fixed (cf. [7]; see also Chapter 18 of [10]).

The organization of this paper is as follows. In Section 2 we show that the length-bounded disjoint paths problem (1) is NP-hard in the strong sense. Next, in Section 3, we consider an auxiliary problem. In Sections 4 we show the pseudo-polynomial time solvability of problem (1) for $k = 2$. Finally, in Section 5 we present a few concluding remarks.

Throughout this paper we shall use the following notation. $V(S), E(S)$ and $G[S]$ represents the set of vertices of G in S, the set edges of G in S and the subgraph of G induced by S, respectively. If P is a path and u, v are vertices in P then $P(u, v)$ denotes the subpath of P connecting u and v. If P is r-t-path and Q is an t-s-path then we denote by $P \cdot Q$ the r-s-path resulting by the concatenation of the paths P and Q. In a directed graph a arc (u, v)-arc is an arc from u to v.

2 NP-hardness

In this section we show that the following problem is NP-hard in the strong sense:

$$\begin{array}{rl} \textit{given}: & \text{– an undirected planar graph } G=(V,E) \text{ embedded in } \mathbb{R}^2; \\ & \text{– } r,s \text{ distinct vertices of } G \text{ adjacent to the unbounded face;} \\ & \text{– a positive integers } b; \\ & \text{– a `length' function } l: E \to \mathbb{Z}_+; \\ \textit{find}: & \text{– pairwise internally vertex-disjoint } r\text{-}s\text{-paths } P_1,\dots,P_k \\ & \quad \text{such that } \mathrm{length}_l(P_i) \le b,\ (i=1,\dots,k). \end{array} \tag{2}$$

This implies that the length bounded disjoint paths problem (1) is NP-hard in the strong sense (even restricted to instances where $b_1 = \cdots = b_k$).

The *3-Satisfiability problem* (3SAT) consists of: *given*: a set of variables $U = \{u_1, \dots, u_n\}$ and a set of three-literal clauses $C = \{c_1, \dots, c_m\}$, each literal been either u_i or $\bar{u}_i$; *question*: is there a satisfying truth assignment for C? A satisfying truth assignment for C is a function $t : U \to \{\text{true}, \text{false}\}$ such that each clause of C contains at least one true literal. 3SAT was shown to be NP-complete by Cook [2].

Theorem 1 *The length-bounded disjoint paths problem (2) is NP-hard in the strong sense.*

Proof. We transform 3SAT to problem (2) for instances where $l(e) \in \{1,2,3\}$ for each $e \in E$.

Let $U = \{u_1, \dots, u_n\}$ be a set of variables and $C = \{c_1, \dots, c_m\}$ be the set of clauses in an arbitrary instance of 3SAT. Corresponding to this instance of 3SAT we define a graph $G = (V,E)$ and a length function $l : E \to \mathbb{Z}_+$ as follows:

$$\begin{array}{rcl} V & := & \{r,s\} \cup \{(i,j) \in \mathbb{Z}_+^2 \mid i \le 4n+2m; j \le 4n \text{ and } i+j \equiv 1 \pmod 2\}; \\ E & := & \{\{(i,j),(k,l)\} \mid \max\{|i-k|,|j-l|\} = 1\} \\ & \cup & \{\{r,(2i-1,0)\} \mid i = 1,\dots,2n+m\} \\ & \cup & \{\{(2i-1,4n),s\} \mid i = 1,\dots,2n+m\}; \end{array}$$

$$l(e) := \begin{cases} 3 & \text{if} \quad e = \{(2i-1,4i-4),(2i-2,4i-3)\} \\ & \text{or} \quad e = \{(2i-1,4i-2),(2i-2,4i-1)\} \\ & \text{or} \quad e = \{(4n+2m-2i+1,4i-4),(4n+2m-2i+2,4i-3)\} \\ & \text{or} \quad e = \{(4n+2m-2i+1,4i-2),(4n+2m-2i+2,4i-1)\} \\ & \qquad \text{for } i = 1,\dots,n; \\ 2 & \text{if} \quad e = \{(2(n+j)-1,4i-4),(2(n+j),4i-3)\} \text{ and } u_i \in c_j \\ & \text{or} \quad e = \{(2(n+j)-1,4i-4),(2(n+j)-2,4i-3)\} \text{ and } \bar{u}_i \in c_j \\ & \qquad \text{for } i = 1,\dots,n; j = 1,\dots,m; \\ 1 & \qquad \text{otherwise} \end{cases}$$

Figure 1 shows the graph associated with

$$C = \{\{\bar{u}_1,\bar{u}_2,u_3\},\{u_1,\bar{u}_2,\bar{u}_3\},\{\bar{u}_1,u_2,\bar{u}_3\},\{u_1,u_2,u_3\}\}.$$

In the figure the number inside the circle next to an edge corresponds to the length of that edge. The length-one edges have no circle close to them on the figure, to avoid overcrowding.

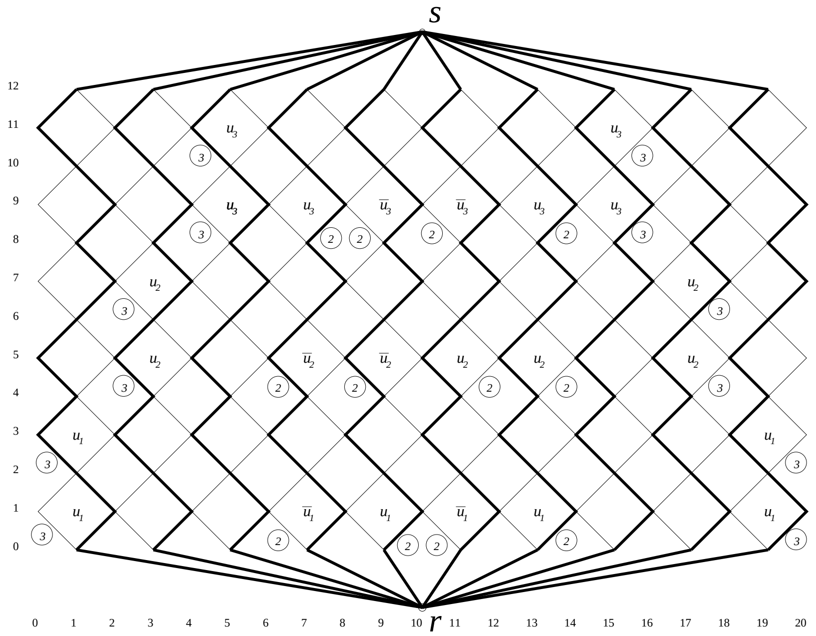

Figure 1: Instance of the maximum length-bounded disjoint paths problem associated with the clause $C = \{\{\bar{u}_1, \bar{u}_2, u_3\}, \{u_1, \bar{u}_2, \bar{u}_3\}, \{\bar{u}_1, u_2, \bar{u}_3\}, \{u_1, u_2, u_3\}\}$.

One can verify that given an instance of 3SAT the corresponding graph G can be constructed in polynomial time and that it can be embedded in $\mathbb{R}^2$ such that r and s are adjacent to the unbounded face. Now, we claim that C is satisfiable if and only if the above constructed graph $G = (V, E)$, the vertices r and s of G, the integer $b := 2n + 4$ and the length function l form a feasible instance of problem (2).

Suppose $P_1, \ldots, P_{2n+m}$ form a feasible solution of problem (2) for the instance constructed above, where P_i is the path containing the vertex $(2i-1, 0)$ $(i = 1, \ldots, 2n+m)$. The path P_{n+j} is the path associated with clause c_j of C $(j = 1, \ldots, m)$ and the paths P_i and $P_{2n+m-i+1}$ are the paths associated with the variable u_i $(i = 1, \ldots, n)$. First we observe that the path P_i must traverse all vertices in $\{(2i-1, j) \in V \mid j = 0, 2, \ldots, 2n\}$ $(i = 1, \ldots, 2n+m)$. Moreover, as each path has length at least $2n+2$ and at most $2n+4$, the subpaths of P_i and $P_{2n+m-i+1}$ connecting $(2i-1, 4i-4)$ to $(2i-1, 4i)$ and $(4n+2m-2i+1, 4i-4)$ to $(4n+2m-2i+1, 4i)$, respectively, must be as shown in either Figure 2(a) or Figure 2(b) $(i = 1, \ldots, n)$. (In the figure the paths are represented by thicker lines.)

We define a truth assignment $t : \{u_1, \ldots, u_n\} \to \{\text{true}, \text{false}\}$ to C as follows

$$t(u_i) := \begin{cases} \text{true} & \text{if paths } P_i((2i-1, 4i-4), (2i-1, 4i)) \text{ and} \\ & \quad P_{2n+m-i+1}((4n+2m-2i+1, 4i-4), (4n+2m-2i+1, 4i)) \\ & \quad \text{are as shown in Figure 2(a);} \\ \text{false} & \text{if paths } P_i((2i-1, 4i-4), (2i-1, 4i)) \text{ and} \\ & \quad P_{2n+m-i+1}((4n+2m-2i+1, 4i-4), (4n+2m-2i+1, 4i)) \\ & \quad \text{are as shown in Figure 2(b).} \end{cases}$$

The truth assignment corresponding to the paths in Figure 1 is $t(u_1) = \text{false}$, $t(u_2) = \text{true}$ and

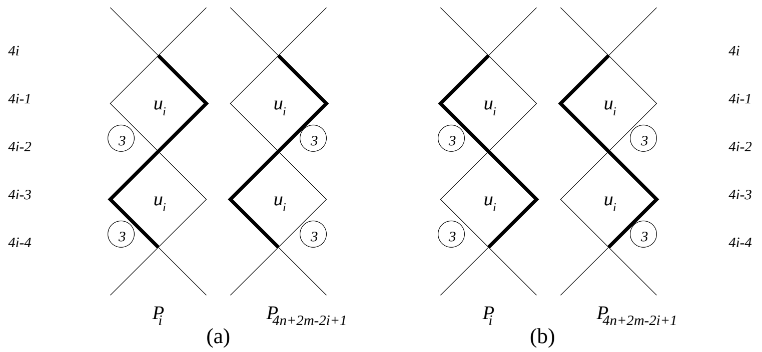

Figure 2: Figure (a) shows the subpaths corresponding to the assignment $t(u_i) :=$ true and Figure (b) shows the subpaths corresponding to the assignment $t(u_i) :=$ false.

$t(u_3) =$ false.

Consider the path P_{n+j} associated with the clause c_j of C. Since P_{n+j} has length at most $2n+4$, this path cannot use the three length-two edges corresponding to the three literals in c_j (length-two edges having a point with first coordinate $2(n+j)-1$ as one of its end vertices). The length-two edge which was not used by P_{n+j} corresponds to a true literal in the clause c_j.

Conversely, if C is satisfiable and t is one of its satisfying truth assignment then we can use $t(u_i)$ to define the subpaths of P_i and $P_{2n+m-i+1}$ connecting $(2i-1, 4i-4)$ to $(2i-1, 4i)$ and $(4n+2m-2i+1, 4i-4)$ to $(4n+2m-2i+1, 4i)$, respectively, as shown in either Figure 2(a) if $t(u_i) =$ true or in Figure 2(b) if $t(u_i) =$ false. There exists a unique way to extend these subpaths to pairwise internally vertex-disjoint r-s-paths $P_1, \ldots, P_{2n+m}$ in G. One can check that each one of these paths has length at most $2n+4$. ∎

The length-bounded disjoint path problem (2) remains NP-hard in the strong sense if one replaces the condition $\text{length}_l(P_i) \leq b$ by $\text{length}_l(P_i) = b$. (Using the same construction as used in Theorem 1 one can transform One-in-three 3SAT (see [3]) to this new problem.) Problem (2) also remains NP-hard in the strong sense if instead of internally vertex-disjoint paths one is interested in edge-disjoint paths. Indeed, we could by "decontracting" each vertex $v \neq r, s$ of degree 4 transform the graph constructed in the theorem in a graph where each vertex $v \neq r, s$ has maximum degree 3.

Itai, Perl and Shiloach [4] have shown that if the condition that r and s are adjacent to a common face is dropped than problem (2) is NP-hard for $k = 2$. To see that for $k = 2$ problem (2) is NP-hard consider the *Partition Problem*: *given*: integers $c_1, \ldots, c_n$; *question*: there exists a set $I \subseteq N := \{1, ..., n\}$ such that $\sum_{i \in I} c_i = \sum_{i \in N \setminus I} c_i$? The Partition Problem was shown to be NP-complete by Karp [5]. Consider the graph G in Figura 3. One can verify that G contains two internally vertex-disjoint r-s-paths each one of length at most $2(n+1) + 1/2 \times \sum_{i=1}^{n} c_i$ if and only if the corresponding instance of the Partition Problem is feasible.

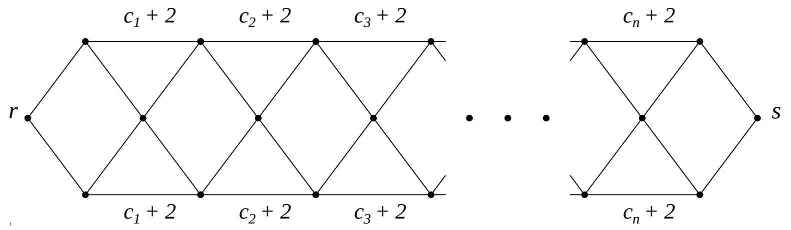

Figure 3: Graph constructed to show that problem (2) is NP-hard for $k = 2$. The edges with no label close to them are length-one edges.

3 An auxiliary problem

In the next sections we reduce problem (1) for $k = 2$ to the following problem:

$$\begin{array}{rl} \textit{given}: & \text{– a directed acyclic graph } D = (W, A); \\ & \text{– distinct vertices } R \text{ and } S \text{ of } D; \\ & \text{– positive integers } b_1, \ldots, b_k; \\ & \text{– length functions } l_1, \ldots, l_k : A \to \mathbb{Z}_+; \\ \textit{find}: & \text{– a directed } R\text{-}S\text{-path } P = (R = v_0, a_1, v_1, \ldots, a_t, v_t = S) \text{ such that} \\ & \quad \text{length}_{l_i}(P) \leq b_i, \quad (i = 1, \ldots, k), \text{ where } \text{length}_{l_i}(P) = \sum_{j=1}^{t} l_i(a_j). \end{array} \tag{3}$$

In this section we observe how problem (3) can be solved in $O(ML^k)$ time and $O(NL^k)$ space, where $N = |W|, M = |A|$ and $L = \max_{1 \leq i \leq k} b_i$. We are assuming a random-access machine with uniform cost measure as model of computation (see [1]). Thus, problem (3) can be solved in pseudo-polynomial for fixed k. For non-fixed k the problem is NP-hard in the strong sense—a reduction very much similar to the reduction shown in the last section proves this.

We use here the same idea used by Li, McCormick and Simchi-Levi [6] in order to find two length-bounded disjoint paths in an acyclic directed graph in pseudo-polynomial time. Namely, we shall keep with each vertex v of D a table T_v with entries indexed by k-tuples $(d_1, \ldots, d_k)$, where $d_i \in \mathbb{Z}$ and $0 \leq d_i \leq L$ $(i = 1, \ldots, k)$. So, T_v has $(L+1)^k$ entries. During the execution of the algorithm an entry $(d_1, \ldots, d_k)$ of T_v contains either nil or an ordered pair $(u, (d'_1, \ldots, d'_k))$ for some $u \in W$ and $(d'_1, \ldots, d'_k) \in \mathbb{Z}^k_+$ such that $(u, v) \in A$ and $d_i = d'_i + l_i(u, v)$ $(i = 1, \ldots, k)$.

Let $v_1, \ldots, v_N$ be an ordering of the vertices of D such that $(v_i, v_j) \in A$ implies $i < j$. This is the so-called *topological ordering* of the vertices of D; it can be found in $O(M)$ time. For our proposes we may assume that $v_1 = R$ and $v_N = S$.

The algorithm begins by setting the entry $(0, \ldots, 0)$ of the table T_R to $(R, (0, \ldots, 0))$ and all others entries of T_R and all entries of all others tables to nil. For $i = 1, \ldots, N$ we iterate the following:

> for each vertex u such that $a := (u, v_i) \in A$ and for each entry $(d_1, \ldots, d_k)$ of T_u with contents different of nil: set the entry $(d_1 + l_1(a), \ldots, d_k + l_k(a))$ of T_{v_i} to $(u, (d_1, \ldots, d_k))$.

If any entry $(d_1, \ldots, d_k)$ of T_S, $(d_i = 0, \ldots, b_i;\ i = 1, \ldots, k)$, has its contents different of nil, say equal to $(u, (d'_1, \ldots, d'_k))$, then there exists an R-S-path $(R = v_0, a_1, v_1, \ldots, v_t, a_t, u, (u, S), S)$ in

D, $v_i \in W$ $(i = 0, \ldots, t)$ $a_i \in A$ $(i = 1, \ldots, t)$, such that $\sum_{j=1}^{t} l_i(a_j) + l_i(u, S) = d'_i + l_i(u, S) = d_i \leq b_i$ $(i = 1, \ldots, k)$. Otherwise the given instance of problem (3) is infeasible.

The algorithm essentially enumerates all possible length vectors $(d_1, \ldots, d_k)$ $(d_i = 0, \ldots, L;\ i = 1, \ldots, k)$, and for each one of them finds a R-S-path P such that $\text{length}_{l_i}(P) = d_i$ for each i, if there exists any. By induction on the number of iterations of the algorithm one can show that at end of iteration i the following holds:

> the entry $(d_1, \ldots, d_k)$ of T_{v_i} contains $(u, (d'_1, \ldots, d'_k))$ for some $u \in W$ and $(d'_1, \ldots, d'_k) \in \mathbb{Z}_+^k$ if and only if there exists a path $(R, a_1, \ldots, a_t, u, (u, v_i), v_i)$ in D such that
>
> $$\sum_{j=1}^{t} l_i(a_j) + l_i(u, v_i) = d'_i + l_i(u, v_i) = d_i \ (i = 1, \ldots, k).$$

This implies the correctness of the algorithm. In order to actually construct an R-S-path at the end of the algorithm we proceed as follows. Suppose the contents of the entry $(d_1, \ldots, d_k)$ of T_S is not nil. Define $v_1 := S$, $d_{1,j} := d_j$ $(j := 1, \ldots, k)$, set i to 1 and iterate the following until $v_i = R$

> increment i and define $v_i := u$; $d_{i,j} := d_{i-1,j} - l_i(u, v_{i-1})$, where u is the vertex in the entry $(d_{i-1,1}, \ldots, d_{i-1,k})$ of $T_{v_{i-1}}$.

Eventually, if $v_{t+1} = R$ then $(R = v_{t+1}, a_t, v_{t-1}, \ldots, a_1, v_1 = S)$, where $a_i = (v_{i+1}, v_i)$ $(i = 1, \ldots, t)$, is an R-S-path satisfying $\sum_{j=1}^{t} l_i(a_j) = d_i$.

The space required by the algorithm is clearly $O(NL^k)$. The amount of time needed to initialize all tables is $O(NL^k)$ and the time necessary to scan vertex v_i during iteration i is $O(d^+(v_i)L^k)$, where $d^+(v_i)$ is the in-degree of v_i. Hence, the overall time required by the algorithm is $O(ML^k)$.

Proposition 2 *Problem (3) can be solved in $O(ML^k)$ time and $O(NL^k)$ space.* ∎

We would like to note that the algorithm gives for a fixed R and for all S, all possible length vectors $(d_1, \ldots, d_k)$ $(d_i = 0, \ldots, L; i = 1, \ldots, k)$, such that there exists a R-S-path P such that $\text{length}_{l_i}(P) = d_i$ for each i. Applying the algorithm N times, we can find for all R and S, all length vectors $(d_1, \ldots, d_k)$ $(d_i = 0, \ldots, L; i = 1, \ldots, k)$, such that there exists a R-S-path P such that $\text{length}_{l_i}(P) = d_i$ for each i. Thus the problem

$$\begin{array}{rl} \textit{given}: & \text{– a directed acyclic graph } D = (W, A); \\ & \text{– positive integers } b_1, \ldots, b_k; \\ & \text{– length functions } l_1, \ldots, l_k : A \longrightarrow \mathbb{Z}_+; \\ \textit{find}: & \text{– for all } R \text{ and } S\ (R, S \in W)\text{, all `shortest' directed } R - S\text{-paths } P \\ & \quad \text{such that } \text{length}_{l_i}(P) \leq b_i\ (i = 1, \ldots, k). \end{array} \tag{4}$$

can be solved in $O(NML^k)$-time and $O(N^2L^k)$-space. Here by a shortest directed R-S-path P is meant a directed R-S-path P such that there is no directed R-S-path P' for which $\text{length}_{l_i}(P') \leq \text{length}_{l_i}(P)$ $(i = 1, \ldots, k)$ and $\text{length}_{l_i}(P') < \text{length}_{l_i}(P)$ for at least one $j \in \{1, \ldots, k\}$.

4 Finding two length-bounded disjoint paths

In this section we show that problems (1) can be solved in pseudo-polynomial time for $k = 2$. The case $k = 1$ can be solved by a single-source shortest path algorithm.

Let us be given a planar graph $G = (V, E)$ embedded in $\mathbb{R}^2$, vertices r and s of G, $b_1, b_2 \in \mathbb{Z}_+$ and a length function $l : E \to \mathbb{Z}_+$. Before proceeding we need some definitions. Let Q be a shortest r-s-path in G and suppose that Q follows the vertices $r = v_0, v_1, \ldots, v_t = s$. For $v, w \in V(Q)$ we write $v \preceq w$ if $v = v_j$ and $w = v_k$ for some $j \leq k$. We write $v \prec w$ if $v \preceq w$ and $v \neq w$. So, $\preceq$ is a partial ordering on $V(Q)$. We denote by R_1 the subgraph of G induced by the vertices in the close region bounded by the path Q and the path on the boundary of G clockwise from s to r. By R_2 we denote the subgraph $G \setminus (V(R_1) \setminus V(Q))$. For $u, v \in V(Q)$ and $i = 1, 2$ let $P(i; u, v)$ be a shortest u-v-path in the graph $R_i \setminus (V(Q) \setminus \{u, v\})$. Figure 4 illustrates the definitions.

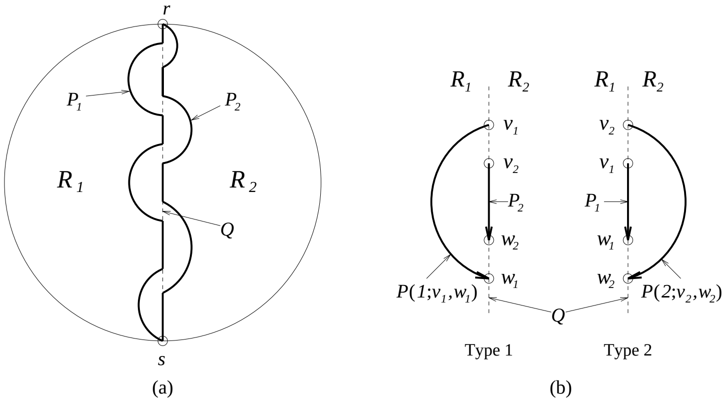

Figure 4: Illustration of the definitions.

Make an auxiliary digraph $D(G, Q) = (W, A)$ with lengths $l_1, l_2, h_1, h_2 : A \to \mathbb{Z}_+$ on its arcs as follows. The vertex set W consists of couples (v_1, v_2) of vertices of Q. In D there is an arc from (v_1, v_2) to (w_1, w_2) if and only if for some $q \in \{1, 2\}$ one of the following conditions hold:

$$\begin{array}{ll} (i) & v_q \prec v_{3-q} \preceq w_{3-q} \prec w_q; \\ (ii) & v_1 = v_2 = r \prec w_{3-q} \prec w_q; \\ (iii) & v_q \prec v_{3-q} \prec w_2 = w_1 = s. \end{array}$$

In addition D has two parallel arcs d_1 and d_2 in D from (r, r) to (s, s). We say that $a \in A$ is an arc of Type t, $t \in \{1, 2\}$ if $a = d_t$ or a fulfills either condition (i), (ii) or (iii) above for $q = t$ (see Figure 4(b)). If $a := ((v_1, v_2), (w_1, w_2)) \in A$ is an arc of Type t then we define:

$$l_t(a) := \text{length}_l(P(t; v_t, w_t)), \quad \text{and} \quad l_{3-t}(a) := \text{length}_l(Q(v_{3-t}, w_{3-t})).$$

The following holds:

Lemma 3 *G has two internally vertex-disjoint r-s-paths P_1 and P_2 such that $\text{length}_l(P_i) \leq b_i$ and P_i is contained in the region R_i $(i = 1, 2)$ if and only if D has an (r, r)-(s, s)-path P such that $\text{length}_{l_i}(P) \leq b_i$ $(i = 1, 2)$.*

Proof. Suppose P_1 and P_2 exist. We may assume that $u, v \in V(P_i) \cap V(Q)$ and $V(P_{3-i}) \cap V(Q(u,v)) = \emptyset$ implies that $P_i(u,v) = Q(u,v)$. (So, we may assume that P_1 and P_2 "look like" the paths in the Figure 4(a).)

Suppose $P_i = Q$ for some $i \in \{1,2\}$. Then $P := ((r,r), d_{3-i}, (s,s))$ is an (r,r)-(s,s)-path in D such that $\mathrm{length}_{l_i}(P) \leq \mathrm{length}_l(P_i) \leq b_i$ $(i = 1,2)$. Thus, we may assume $P_i \neq Q$ $(i = 1,2)$. Let $\{v_{1,1}, v_{2,1}\}, \ldots, \{v_{1,d-1}, v_{2,d-1}\}$ be pairs of end points of maximal subpaths of Q whose internal vertices are used neither by P_1 nor by P_2. We may assume $v_{1,1} \prec v_{2,1} \preceq v_{2,2} \prec v_{1,2} \preceq v_{1,3} \prec v_{2,3} \ldots$. Let $a_i := ((v_{1,i-1}, v_{2,i-1}), (v_{1,i}, v_{2,i}))$ $(i = 1, \ldots, d)$, where $(v_{1,0}, v_{2,0}) = (r,r)$ and $(v_{1,d}, v_{2,d}) = (s,s)$. One can check that $a_i \in A$ $(i = 1, \ldots, d)$ and $P := ((r,r), a_1, \ldots, a_d, (s,s))$ is an (r,r)-(s,s)-path in D fulfilling the conditions of the lemma.

Conversely, suppose

$$P := ((r,r) = (v_{1,0}, v_{2,0}), a_1, (v_{1,1}, v_{2,1}), \ldots, a_d, (v_{1,d}, v_{2,d}) = (s,s))$$

is an (r,r)-(s,s)-path in D such that $\mathrm{length}_{l_i}(P) \leq b_i$ $(i = 1,2)$. For $i = 1,2$ and $j = 1, \ldots, d$ let

$$P_{i,j} \; := \; \begin{cases} P(i; v_{i,j-1}, v_{i,j}) & \text{if } a_j \text{ is an arc of Type } (i); \\ Q(v_{i,j-1}, v_{i,j}) & \text{otherwise.} \end{cases}$$

We define $P_i := P_{i,1} \cdot \cdots \cdot P_{i,d}$ $(i = 1,2)$. One can verify that P_1, P_2 as defined are internally vertex-disjoint r-s-paths (not necessarily simple) such that $\mathrm{length}_l(P_i) = \mathrm{length}_{l_i}(P) \leq b_i$ and P_i is contained in the region R_i $(i = 1,2)$. ∎

Theorem 4 *One can find in pseudo-polynomial time two internally vertex-disjoint r-s-paths P_1 and P_2 of G (if there exist any) such that $\mathrm{length}_l(P_i) \leq b_i$ and P_i is contained in the region R_i $(i = 1,2)$.*

Proof. By solving two all-pairs shortest path problem we can compute $\{P(i; u,v) \mid u,v \in V(Q) \text{ and } i = 1,2\}$. (For all pairs of vertices $u,v \in V(Q)$ find a shortest u-v-path in the graph $R_i \setminus (V(Q) \setminus \{u,v\})$ $(i = 1,2)$.) Thus, the digraph D can be construct in polynomial time. D is acyclic as $((v_1,v_2),(w_1,w_2)) \in A$ implies that (w_1,w_2) lexicographically greater than (v_1,v_2) (with respect to $\preceq$). Hence, it follows from Proposition 2 that an (r,r)-(s,s)-path P such that $\mathrm{length}_{l_i}(P) \leq b_i$ and P_i is contained in the region R_i $(i = 1,2)$ can be found (if there exists any) in $O(|V(Q)|^4 L^2)$ time and $O(|V(Q)|^2 L^2)$ space, where $L := \max\{b_1, b_2\}$. Now, the theorem follows from Lemma 3. ∎

Corollary 5 *The length-bounded disjoint path problem (1) can be solved in pseudo-polynomial time for $k = 2$.*

Proof. Let us be given a instance of problem (1) consisting of: a graph $G' = (V', E')$; pairs $\{r_1, s_1\}$ and $\{r_2, s_2\}$ of vertices of G'; positive integers b_1 and b_2 and a length function $l' : E' \to \mathbb{Z}_+$. We define a graph $G = (V,E)$ by adding to G' two new vertices r and s and the four edges $\{r, r_1\}, \{r, r_2\}, \{s, s_1\}, \{s, s_2\}$. We define the length function $l : E \to \mathbb{Z}_+$ such that $l(e) = l'(e)$ if $e \in E'$ and $l(e) = 0$ for the new edges. The given instance of problem (1) is feasible if and only if G posses internally vertex-disjoint r-s-paths P_1 and P_2 such that $\mathrm{length}_l(P_i) \leq b_i$ and P_i is contained in the region R_i $(i = 1,2)$. ∎

From Theorem 4 one can also derive that the length-bounded disjoint path problem (2) can be solved in pseudo-polynomial time for $k = 2$ even if instead of seeking for two r-s-paths of length bounded by b we are seeking for a two r-s-paths one of length bounded by a positive integer b_1 and one of length bounded by a positive integer b_2.

5 Concluding remarks

We would like to make a few final remarks. In this paper it has been shown that problem (1) is NP-hard in the strong sense and a pseudo-polynomial time algorithm has been given for finding two length-bounded disjoint paths. We mention that using a similar technique as presented in Section 4 one can show that problem (1) can be solved in pseudo-polynomial time for $k = 3$ and $r_1, r_2, r_3, s_3, s_2, s_1$ occurring in this order when following the boundary of G (say) clockwise. Unfortunately, the proof is tedious and unattractive, so we have chosen to omit it.

In a forthcoming paper, applying methods of the present paper, we show the pseudo-polynomial time solvability of the length-bounded disjoint path problem (1) for the following special case:

> k is fixed and the vertices $r_1, s_1, \ldots, r_k, s_k$ occur in this order when following the boundary of G clockwise.

Acknowledgments

We thank Bert Gerards and Alexander Schrijver for their many insightful remarks and explanations.

References

[1] A.V. Aho, J.E. Hopcroft, and J.D. Ullman, *The design and analysis of computer algorithms*, Addison Wesley, reading, Mass., 1994.

[2] S.A. Cook, *The complexity of theorem-proving procedures*, Proceedings of Third ACM Symposium on Theory of Computing (New York), Association for Computing Machinery, 1971, Shaker Heights, Ohio, 1971, pp. 151–158.

[3] M.R. Garey and D.S. Johnson, *Computer and intractability: A guide to the theory of NP-completness*, Freeman, San Francisco, 1979.

[4] I. Itai, Y. Perl, and Y. Shiloach, *The complexity of finding maximum disjoint paths with length constraints*, Networks **12** (1982), 277–286.

[5] R.M. Karp, *Reducibility among combinatorial problems*, Complexty of Computer Computatios (New York) (R.E. Miller and J.W. Thatcher, eds.), Plenum Press, 1972, pp. 85–104.

[6] C. Li, S.T. McCormick, and D. Simchi-Levi, *The complexity of finding paths with min-max objective function*, Discrete Applied Mathematics **26** (1990), 105–115.

[7] C.H. Papadimitriou, *On the complexity of integer programming*, Journal of the Association for Computing Machinery **28** (1981), 765–768.

[8] C.H. Papadimitriou and K. Steiglitz, *Combinatorial optimization: Algorithms and complexity*, Prentice-Hall, Englewood Cliffs, N.J., 1982.

[9] 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.

[10] A. Schrijver, *Theory of linear and integer programming*, John Wiley and Sons, Chichester, 1986.

[11] L.A. Wolsey, *A view of shortest route methods in integer programming*, Cahiers du Centre d'Études de Recherche Opérationelle **16** (1974), 317–335.