Min-Sum 2-Paths Problems

Trevor Fenner[*] Oded Lachish[†] Alexandru Popa[‡]

Abstract

An *orientation* of an undirected graph G is a directed graph obtained by replacing each edge $\{u, v\}$ of G by exactly one of the arcs (u, v) or (v, u). In the *min-sum k-paths orientation problem*, the input is an undirected graph G and ordered pairs (s_i, t_i), where $i \in \{1, 2, \ldots, k\}$. The goal is to find an orientation of G that minimizes the sum over every $i \in \{1, 2, \ldots, k\}$ of the distance from s_i to t_i.

In the *min-sum k edge-disjoint paths problem* the input is the same, however the goal is to find for every $i \in \{1, 2, \ldots, k\}$ a path between s_i and t_i so that these paths are edge-disjoint and the sum of their lengths is minimum. Note that, for every fixed $k \geq 2$, the question of **NP**-hardness for the min-sum k-paths orientation problem and the min-sum k edge-disjoint paths problem have been open for more than three decades. We study the complexity of these problems when $k = 2$.

We exhibit a PTAS for the min-sum 2-paths orientation problem. A by-product of this PTAS is a reduction from the min-sum 2-paths orientation problem to the min-sum 2 edge-disjoint paths problem. The implications of this reduction are: (i) an **NP**-hardness proof for the min-sum 2-paths orientation problem yields an **NP**-hardness proof for the min-sum 2 edge-disjoint paths problem, and (ii) any approximation algorithm for the min-sum 2 edge-disjoint paths problem can be used to construct an approximation algorithm for the min-sum 2-paths orientation problem with the same approximation guarantee and only an additive polynomial increase in the running time.

1 Introduction

In communications, *Multihoming* is the process of communicating through more than one connection. The goal is to increase communication reliability. Now imagine that each connection must be made between two distinct entities, for example, if a customer has numerous internet providers, each with a distinct entry point that requires a connection to a distinct end-point, see [1, 8]. This is the case we deal with here.

In order to optimize reliability when using multiple connections a natural goal is that the channels are disjoint. We model the problem of determining whether such channels exist with the *k edge-disjoint paths problem*, where the input is an instance consisting of a graph and pairs of

[*]Birkbeck, University of London, London, UK. Email: trevor@dcs.bbk.ac.uk

[†]Birkbeck, University of London, London, UK. Email: oded@dcs.bbk.ac.uk

[‡]Department of Communications and Networking Aalto University School of Electrical Engineering P.O. Box 13000, 00076 Aalto, FINLAND. Email: alexandru.popa@aalto.fi

vertices $\{s_i, t_i\}$, where $i \in \{1, 2, \ldots, k\}$, and the goal is to find k edge-disjoint paths between the k pairs $\{s_i, t_i\}$. Robertson and Seymour proved in [7] that, for fixed k, this problem is in $\mathbf{P}$.

However, just having k edge-disjoint paths is often not sufficient. A natural requisite is that the paths found are optimized according to some condition. Such conditions can be minimum maximal length or minimum sum of lengths. These conditions lead to two optimization problems: the first is known as the *min-max k edge-disjoint paths problem*; and the latter as the *min-sum k edge-disjoint paths problem*. In [6] Li et al. show that the min-max k edge-disjoint paths problem is $\mathbf{NP}$-hard, even when $k = 2$ and $\{s_1, t_1\} = \{s_2, t_2\}$. In contrast, the question of $\mathbf{NP}$-hardness of the min-sum k edge-disjoint paths problem for fixed $k \geq 2$ has been open for more than twenty years.

An *orientation* of an undirected graph G is a directed graph obtained by replacing each edge $\{u, v\}$ of G by exactly one of the arcs (u, v) or (v, u). In the *min-sum k-paths orientation problem*, the input instance is an undirected graph G and ordered pairs (s_i, t_i), where $i \in \{1, 2, \ldots, k\}$. The goal is to find an orientation of G in which the sum over all $i \in \{1, 2, \ldots, k\}$ of the distance from s_i to t_i is minimized. The min-sum k-paths orientation problem is a relaxation of the min-sum k edge-disjoint paths problem in the following sense: if the requirement for a path between s_i and t_i for each $i \in \{1, 2, \ldots, k\}$ is replaced by the requirement for an unsplittable flow of size 1 from s_i to t_i for each $i \in \{1, 2, \ldots, k\}$ and these flows may share edges if they are in the same direction, then we get the min-sum k-paths orientation problem. We note that the question of $\mathbf{NP}$-hardness for the min-sum k-paths orientation problem, for fixed $k \geq 2$, has also been open for more than twenty years. In this paper we focus on the min-sum 2-paths orientation problem and its relation with the min-sum 2 edge-disjoint paths problem.

There have been a number of results for the min-sum k edge-disjoint paths problem. Zhang and Zhao [10] have shown that in general graphs for general k the min-sum k edge-disjoint paths problem is FP^{NP}-complete. They gave a bicriteria approximation algorithm for the problem. There have also been a number of results for the min-sum 2 edge-disjoint paths problem. Zhang and Zhao have shown that this problem has a constant factor approximation. Kobayashi and Sommer [5] showed that the problem is in $\mathbf{P}$ if G is planar and s_1, t_1, s_2 and t_2 are on at most two faces of the graph. Kammer et al. [4] showed that it is in $\mathbf{P}$ if G is a chordal graph. For a comprehensive discussion of results, see Kobayashi and Sommer [5].

Finally, the min-sum k-paths orientation problem has been studied by Hassin and Megiddo [2]. There they showed that this problem is $\mathbf{NP}$-hard for general k. They also studied the *min-max k paths-orientation problem*. They proved that this problem is $\mathbf{NP}$-hard even for $k = 2$. In [3], Ito et al. also studied these two problems. They showed that, for unrestricted k, the min-sum k-paths orientation problem does not have a polynomial time algorithm with an approximation factor of 2 or less, unless $\mathbf{P} = \mathbf{NP}$. They presented approximation algorithms for restricted variations of this problem, for example, for certain classes of graphs, such as cacti.

In this paper, we exhibit a PTAS for the min-sum 2-paths orientation problem. A by-product of this PTAS is a reduction from the min-sum 2-paths orientation problem to the min-sum 2 edge-disjoint paths problem. The implications of this reduction are: (i) that an $\mathbf{NP}$-hardness proof for the min-sum 2-paths orientation problem yields an $\mathbf{NP}$-hardness proof for the min-sum 2 edge-disjoint paths problem, and (ii) that any approximation algorithm for the min-sum 2 edge-disjoint paths problem can be used to construct an approximation algorithm for the min-sum 2-paths orientation problem with the same approximation guarantee and only an additive polynomial increase in the running time. Our results suggest that if indeed the min-sum 2-paths orientation problem is **NP**-hard, then proving this may be more difficult than it seems because of the implication for the min-sum 2 edge-disjoint paths problem. The reduction also implies, according to results by Kobayashi and Sommer [5] and Kammer et al. [4] for the min-sum 2 edge-disjoint paths problem, that the orientation problem is in **P** if G is chordal or if it is planar and s_1, t_1, s_2 and t_2 are on at most two faces of the graph.

One of the central ingredients we use is a structural lemma that states that for any given input instance (G, s_1, t_1, s_2, t_2), if there exists an orientation in which the distances from s_1 to t_1 and from s_2 to t_2 are both finite there exists an optimal orientation with two min-sum directed paths, one from s_1 to t_1 and the other from s_2 to t_2, such that either (i) these directed paths are arc-disjoint, or (ii) the directed paths are not arc-disjoint and their common edges form a directed-path. We obtain the reduction to the min-sum 2 edge-disjoint paths problem by showing that if, on the same input instance, we execute an algorithm for min-sum 2 edge-disjoint problem and an algorithm that works if (ii) holds, then the best result is optimal. We obtain the PTAS in a similar manner, by showing that a PTAS exists for instances on which (i) holds.

2 Preliminaries

We use $[k]$ to denote the set $\{1, 2, \dots, k\}$. An undirected *graph* is an ordered pair $G = (V, E)$, where V is a set of *vertices* and E is a set of *edges*, each edge being a subset of V of size two. A *directed graph* is an ordered pair $\vec{G} = (V, \vec{E})$, where V is a set of vertices and $\vec{E}$ is a set of ordered pairs of vertices of V called *arcs*. We use the notation $V(G)$ for the set of vertices of G or $\vec{G}$ and $E(G)$ for the set of edges of G, and $E(\vec{G})$ for the set of arcs of $\vec{G}$. When clear from the context we use n instead of $|V(G)|$.

Definition 1 [Orientation] *An* ***orientation*** *of an undirected graph $G = (V, E)$ is a directed graph $\vec{H} = (V, \vec{E})$ such that, for every $\{u, v\} \in E$, either $(u, v) \in \vec{E}$ or $(v, u) \in \vec{E}$, but not both. We use the notation $\vec{H}_G$ to denote that $\vec{H}$ is an orientation of G.*

A *path* P or a *dipath* $\vec{P}$ in G or $\vec{G}$, respectively, is a tuple $(u_1, u_2, \dots, u_k) \in V^k$ such that for every $i \in [k-1]$ we have that $\{u_i, u_{i+1}\} \in E(G)$ or $(u_i, u_{i+1}) \in E(\vec{G})$, respectively, and $u_1, u_2, \dots, u_k$ are all distinct. The path $(u, \dots, v)$ in G is a path **between** u **and** v. The dipath $(u, \dots, v)$ in $\vec{G}$ is a dipath **from** u **to** v. We use the notation $P_{u,v}$ to indicate that the path is between u and v, and the notation $\vec{P}_{u,v}$ to indicate that the dipath is from u to v. A *cycle* in G is a tuple $C = (u_1, u_2, \dots, u_k, u_1) \in V^{k+1}$ such that $(u_1, u_2, \dots, u_k)$ is a path and $\{u_k, u_1\} \in E(G)$. Note that we often consider a path to be a subgraph.

A path $P' = (u_1, \dots, u_\ell)$ in a graph is a *subpath* of the path $P = (v_1, \dots, v_k)$ if there exists $i \in [k - \ell + 1]$ such that $(u_1, u_2, \dots, u_\ell) = (v_i, v_{i+1}, \dots, v_{i+\ell-1})$. A graph (V', E') is a *subgraph* of a graph (V, E) if $V' \subseteq V$ and $E' \subseteq E$.

The *length* of P or $\vec{P}$, denoted by $len(P)$ or $len(\vec{P})$, respectively, is $k-1$. The *distance* between u and v in $V(G)$, denoted by $dist_G(u, v)$, is the length of a shortest path between u and v if such a path exists, and $dist_G(u, v) = \infty$ otherwise. The *distance* between a pair of paths P and P' in G, denoted by $dist_G(P, P')$, is the minimal distance between a vertex in $V(P)$ and a vertex in $V(P')$.

The *distance* from $u \in V(G)$ to $v \in V(G)$ in a directed graph $\vec{G}$, denoted by $dist_{\vec{G}}(u, v)$, is the length of a shortest dipath from u to v if such a dipath exists, and $dist_{\vec{G}}(u, v) = \infty$ otherwise. When the graph under consideration is clear from context, we simply write $dist(u, v)$.

Definition 2 $[B_G(v, x)]$ *Let G be a graph, $v \in V(G)$ and $x > 0$. Then $B_G(v, x)$ is the subset of $E(G)$ containing all the edges $\{u, w\} \in E(G)$ such that $dist_G(v, u) < x$ and $dist_G(v, w) < x$.*

Definition 3 **[Instance]** *An **instance** is an ordered tuple (G, s_1, t_1, s_2, t_2) such that G is an undirected graph and s_1, t_1, s_2 and t_2 are vertices in $V(G)$.*

Problem 4 (Min-Sum 2 Edge-Disjoint Paths) *Given an instance (G, s_1, t_1, s_2, t_2), find edge disjoint paths P_{s_1,t_1} and P_{s_2,t_2} such that $len(P_{s_1,t_1}) + len(P_{s_2,t_2})$ is minimum.*

2.1 The Min-Sum *2* Paths Orientation Problem

Problem 5 (Min-Sum 2 Paths Orientation) *Given an instance (G, s_1, t_1, s_2, t_2), find an orientation $\vec{H}_G$ of G that minimizes $dist_{\vec{H}_G}(s_1, t_1) + dist_{\vec{H}_G}(s_2, t_2)$. We call such an orientation an **optimal orientation**.*

Definition 6 $[OPT]$ *Let (G, s_1, t_1, s_2, t_2) be an instance. We define $OPT(G, s_1, t_1, s_2, t_2) = dist_{\vec{H}_G}(s_1, t_1) + dist_{\vec{H}_G}(s_2, t_2)$ for any optimal orientation $\vec{H}_G$. We write OPT when the instance under consideration is clear from the context.*

We make the following definition in order to recast the problem in terms of undirected graphs.

Definition 7 **[Non-conflicting paths and optimal paths]** *Let G be an undirected graph and $x_1, y_1, x_2, y_2 \in V(G)$. Paths P_{x_1,y_1} and P_{x_2,y_2} in G are **non-conflicting** if there exists an orientation $\vec{H}_G$ in which $\vec{P}_{x_1,y_1} = P_{x_1,y_1}$ and $\vec{P}_{x_2,y_2} = P_{x_2,y_2}$ and are **optimal** if they are non-conflicting and $len(P_{x_1,y_1}) + len(P_{x_2,y_2}) = OPT(G, x_1, y_1, x_2, y_2)$ for the instance (G, x_1, y_1, x_2, y_2).*

Observe that for any optimal orientation $\vec{H}_G$ for an instance (G, s_1, t_1, s_2, t_2) any two shortest dipaths $(s_1, \ldots, t_1)$ and $(s_2, \ldots, t_2)$ in $\vec{H}_G$ are an optimal pair of paths and in particular a non-conflicting pair of paths. We note that checking whether two paths are non-conflicting can easily be done in polynomial time. By the following observation, we see that, in order to show that $OPT(G, s_1, t_1, s_2, t_2) \leq k$, it is sufficient to find non-conflicting paths P_{s_1,t_1} and P_{s_2,t_2} such that $len(P_{s_1,t_1}) + len(P_{s_2,t_2}) \leq k$.

Observation 8 *Let (G, s_1, t_1, s_2, t_2) be an instance. If P_{s_1,t_1} and P_{s_2,t_2} are non-conflicting, then $OPT(G, s_1, t_1, s_2, t_2) \leq len(P_{s_1,t_1}) + len(P_{s_2,t_2})$.*

Without loss of generality, we always make the following assumption:

Assumption 9 *For every given instance (G, s_1, t_1, s_2, t_2), we assume that $OPT < \infty$, G is connected and that s_1, t_1, s_2, t_2 are distinct.*

We may make this assumption since it is easy to decide whether $OPT = \infty$ and the problem on an instance (G, s_1, t_1, s_2, t_2) such that s_1, t_1, s_2 and t_2 are not distinct can be easily reduced to the problem on an instance $(G', s'_1, t'_1, s'_2, t'_2)$ where s'_1, t'_1, s'_2 and t'_2 are distinct.

3 Algorithm Overview and Definitions

We start by giving an algorithm that finds an optimal pair of paths for a restricted set of instances. Afterwards we explain how to obtain our claimed results by extending this algorithm.

Let (G, s_1, t_1, s_2, t_2) be an instance that has an optimal pair of edge-disjoint paths P_{s_1,t_1} and P_{s_2,t_2} such that $dist(P_{s_1,t_1}, P_{s_2,t_2}) > dist(s_1, t_1)/2$. Consequently, any shortest path between s_1 and t_1 does not intersect P_{s_2,t_2}. For such an instance finding an optimal pair of paths can be done as follows: (i) find a shortest path P'_{s_1,t_1} (ii) let G' be the graph resulting from removing the edges of P'_{s_1,t_1} from G, and (iii) find a shortest path P'_{s_2,t_2} in G'. We refer to this as the *simple algorithm*.

Observe that G' contains all the edges of P_{s_2,t_2}, since P'_{s_1,t_1} and P_{s_2,t_2} are edge disjoint. Hence, $len(P'_{s_2,t_2}) \leq len(P_{s_2,t_2})$. Since P'_{s_1,t_1} is also a shortest path $len(P'_{s_1,t_1}) \leq len(P_{s_1,t_1})$. Consequently, P'_{s_1,t_1} and P'_{s_2,t_2} are an optimal pair of paths.

We have demonstrated that, if an instance has optimal pair that are sufficiently far from each other, then the problem of finding an optimal pair of paths requires only polynomial time. The distance between the paths of an optimal pair is crucial for our results. Hence, we make the following definition.

Definition 10 [$\Delta(G, s_1, t_1, s_2, t_2)$ **and** $\delta(G, s_1, t_1, s_2, t_2)$] *Let (G, s_1, t_1, s_2, t_2) be an instance. We define $\Delta(G, s_1, t_1, s_2, t_2)$ and $\delta(G, s_1, t_1, s_2, t_2)$ to be the maximum and minimum, respectively, of $dist_G(P_{s_1,t_1}, P_{s_2,t_2})/OPT$ over all optimal pairs of paths P_{s_1,t_1} and P_{s_2,t_2}. We write just Δ and δ when the instance under consideration is clear from the context.*

Obviously, $\delta \leq \Delta$. Note that if $\Delta(G, s_1, t_1, s_2, t_2) > 1/2$, then we can use the simple algorithm to find an optimal pair of paths. For our results we need something stronger. We next describe an algorithm, similar in essence to the simple algorithm, which for input $\epsilon > 0$ and instance (G, s_1, t_1, s_2, t_2) finds an optimal pair of paths in time $n^{O(1/\epsilon)}$ if $\epsilon < \Delta$. Our final algorithm is a slight variation of this.

Let $\epsilon > 0$ such that $\epsilon < \Delta$ and suppose that P_{s_1,t_1} and P_{s_2,t_2} are an optimal pair of paths $\Delta \cdot OPT$ apart. Suppose also that we have a set of $h = O(1/\epsilon)$ vertices $u_1, u_2, \ldots, u_h \in V(P_{s_1,t_1})$, where $u_1 = s_1$, $u_h = t_1$ and $dist_{P_{s_1,t_1}}(u_i, u_{i+1}) < \epsilon \cdot OPT$ for each $i \in [h-1]$. Now apply the following algorithm, which we call the *basic algorithm*: (i) find a shortest path P'_{s_1,t_1} in the graph $\Big(V(G), \bigcup_{i\in[h]} B_G(u_i, \epsilon \cdot OPT)\Big)$, (ii) find a shortest path P'_{s_2,t_2} in the graph $\Big(V(G), E(G) \setminus E(P'_{s_1,t_1})\Big)$. We show that P'_{s_1,t_1} and P'_{s_2,t_2} are an optimal pair of paths.

First observe that all the edges of P_{s_1,t_1} are contained in $\bigcup_{i\in[h]} B_G(u_i, \epsilon \cdot OPT)$ and hence $len(P'_{s_1,t_1}) \leq len(P_{s_1,t_1})$. Since $\epsilon < \Delta$, by Definition 10, P_{s_2,t_2} and $B_G(u_i, \epsilon \cdot OPT)$ are edge-disjoint for each $i \in [h]$. Thus, P'_{s_1,t_1} and P_{s_2,t_2} are also edge-disjoint. It follows, from the simple algorithm, that P'_{s_1,t_1} and P'_{s_2,t_2} are an optimal pair of paths.

Therefore for the rest of this section we assume that $\epsilon \geq \Delta$. In order to deal with this case, we now prove a structural result that states that any non-trivial instance is of at least one of the following two types.

Definition 11 [**Disjoint Instance and Intersecting Instance**] *An instance (G, s_1, t_1, s_2, t_2) is* ***disjoint*** *if it has an optimal pair of paths P_{s_1,t_1} and P_{s_2,t_2} that are edge-disjoint. An instance*

(G, s_1, t_1, s_2, t_2) *is **intersecting** if it has an optimal pair of paths* P_{s_1,t_1} *and* P_{s_2,t_2} *that are not edge-disjoint and whose common edges form a subpath of both* P_{s_1,t_1} *and* P_{s_2,t_2}*.*

We prove the following lemma in Appendix 6.

Lemma 12 *Let* (G, s_1, t_1, s_2, t_2) *be an instance for which* $OPT < \infty$*, then* (G, s_1, t_1, s_2, t_2) *is disjoint or intersecting (or both).*

Suppose that P_{s_1,t_1} and P_{s_2,t_2} are an optimal pair of paths that either: (i) are edge-disjoint and $dist(P_{s_1,t_1}, P_{s_2,t_2}) = \Delta \cdot OPT$, or (ii) all their common edges form a path P_{m_0,m_1}. In case (i) let m_0 and m_1 be such that $dist(m_0, m_1) = \delta \cdot OPT$ where m_0 is on one of the paths P_{s_1,t_1} and P_{s_2,t_2} and m_1 is on the other. We note that $m_0 = m_1$ if $\delta = 0$. Since there are less than n^2 potential pairs, we can assume we have one since we can try each pair in turn. The advantage of knowing such a pair $\{m_0, m_1\}$ is that every shortest path between m_0 and m_1 is edge disjoint from both P_{s_1,t_1} and P_{s_2,t_2}. We call such a pair a *pivot.* More formally:

Definition 13 ***[Pivot]*** *Let* (G, s_1, t_1, s_2, t_2) *be an instance. A **pivot** is a pair* $\{m_0, m_1\}$ *such that one of the following holds:*

1. (G, s_1, t_1, s_2, t_2) *is disjoint and* $dist_G(m_0, m_1) = \delta \cdot OPT$ *for some optimal pair of edge-disjoint paths* P_{s_1,t_1} *and* P_{s_2,t_2}*, where* m_0 *is in one of these paths and* m_1 *is in the other, or*

2. (G, s_1, t_1, s_2, t_2) *is intersecting with optimal paths whose common edges form a path* P_{m_0,m_1}*.*

If Case 1 holds, then $\{m_0, m_1\}$ *is a **disjoint-pivot**, and if Case 2 holds, then* $\{m_0, m_1\}$ *is an **intersecting-pivot**. In both cases* P_{m_0,m_1} *is necessarily a shortest path.*

Let $\{m_0, m_1\}$ be a pivot and P_{m_0,m_1} be a shortest path. The naive way of proceeding is to use a min-cost single source flow algorithm for a flow of size 4 as follows: (i) let G' be obtained from G by adding a vertex a and four edges: two between a and m_0 and two between a and m_1; (ii) solve the min-cost single source flow with a being the source, s_1, t_1, s_2 and t_2 being the targets, and all edges of G' having capacity and cost 1. This will result in four min-sum edge-disjoint paths P_{x_1,m_0}, P_{x_2,m_0}, P_{x_3,m_1}, P_{x_4,m_1}, where $\{x_1, x_2, x_3, x_4\} = \{s_1, t_1, s_2, t_2\}$. Now the natural conjecture is that a non-conflicting pair of paths as required can be found in the graph consisting of all vertices and edges of these four paths and of P_{m_0,m_1}. However, this idea does not work if the configuration obtained is like that in Figure 1. Thus, a different strategy is required. The strategy we use in

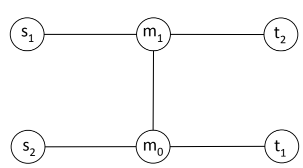


Figure 1: [Naive Attempt]

Section 4 is to first find a min-sum edge-disjoint pair P_{s_1,m_i}, $P_{t_2,m_{1-i}}$, where $i \in \{0,1\}$ and then a min-sum edge-disjoint pair P_{s_2,m_j}, $P_{t_1,m_{j-i}}$, where $j \in \{0,1\}$. We shall show that the graph consisting of the vertices and edges of these four paths and of P_{m_0,m_1} is sufficient for finding the required non-conflicting pair of paths.

In Section 5, we introduce the algorithm that works when $\epsilon < \Delta$ and in Section 6 we prove the main results.

4 Algorithm 1

We introduce here the algorithm for the case that Δ is small ($\Delta \leq \epsilon$) and hence δ is even smaller.

Algorithm 1

Input: instance (G, s_1, t_1, s_2, t_2)

- Iterate over all pairs of vertices $\{m_0, m_1\} \subseteq V$

  1. $P_{m_0,m_1} \longleftarrow$ an arbitrary shortest path between m_0 and m_1
  2. $G' \longleftarrow (V(G), E(G) \setminus E(P_{m_0,m_1}))$
  3. In G', find min-sum edge-disjoint paths P'_{s_1,m_i} and $P'_{t_2,m_{1-i}}$, where $i \in \{0,1\}$
  4. In G', find min-sum edge-disjoint paths P'_{s_2,m_j} and $P'_{t_1,m_{1-j}}$, where $j \in \{0,1\}$
  5. Define Q to be the undirected graph such that
     (a) $V(Q) = V(P'_{s_1,m_i}) \cup V(P'_{t_2,m_{1-i}}) \cup V(P'_{s_2,m_j}) \cup V(P'_{t_1,m_{1-j}}) \cup V(P_{m_0,m_1})$
     (b) $E(Q) = E(P'_{s_1,m_i}) \cup E(P'_{t_2,m_{1-i}}) \cup E(P'_{s_2,m_j}) \cup E(P'_{t_1,m_{1-j}}) \cup E(P_{m_0,m_1})$
  6. Using the method explained in Lemma 15, find non-conflicting paths $P^{m_0,m_1}_{s_1,t_1}$ and $P^{m_0,m_1}_{s_2,t_2}$ in Q such that

$$len(P^{m_0,m_1}_{s_1,t_1}) + len(P^{m_0,m_1}_{s_2,t_2}) \leq$$
$$len(P'_{s_1,m_i}) + len(P'_{t_2,m_{1-i}}) + len(P'_{s_2,m_j}) + len(P'_{t_1,m_{1-j}}) + 2 \cdot len(P_{m_0,m_1})$$

Output: The paths $P^{m_0,m_1}_{s_1,t_1}$ and $P^{m_0,m_1}_{s_2,t_2}$ that minimize $len(P^{m_0,m_1}_{s_1,t_1}) + len(P^{m_0,m_1}_{s_2,t_2})$

Theorem 14 *Let $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ be the paths returned by Algorithm 1 on instance (G, s_1, t_1, s_2, t_2). Then $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ are non-conflicting and $len(P^*_{s_1,t_1}) + len(P^*_{s_2,t_2}) \leq (1 + 2\delta) \cdot OPT$. The running time of Algorithm 1 is bounded by a polynomial function of n.*

[Note that if the input instance is intersecting, then $\delta = 0$ and hence Algorithm 1 returns an optimal pair of paths.]

Proof. Finding the paths in Steps 3 and 4 can be done by reducing the problem to finding edge-disjoint paths from a single vertex as follows. Add to the graph G', computed in Step 2, a vertex a and edges $\{a, m_0\}$ and $\{a, m_1\}$. Then find a pair of min-sum edge-disjoint paths each from a to s_1 and t_2 in Step 3 and s_2 and t_1 in Step 4. According to Yang et al. [9] this requires a running time of $O(n^2)$. By Lemma 15 below, Step 6 requires a running time that is polynomial in n. Since all other steps also require at most a polynomial in n running time and there are fewer than n^2 iteration of Steps 1 to 6, the overall running time is polynomial in n.

Suppose that $\{m_0, m_1\}$ is a pivot with associated optimal pair of paths P_{s_1,t_1} and P_{s_2,t_2} and that P_{m_0,m_1} is the shortest path found in Step 1. Let $\xi = len(P'_{s_1,m_i}) + len(P'_{t_2,m_{1-i}}) + len(P'_{s_2,m_j}) + len(P'_{t_1,m_{1-j}}) + 2 \cdot len(P'_{m_0,m_1})$.

As noted earlier, when $\{m_0, m_1\}$ is a disjoint-pivot, then P_{m_0,m_1} does not share any edges with P_{s_1,t_1} and P_{s_2,t_2}. Consequently, the paths found Step 3 and 4 have overall at most OPT edges. Hence, $\xi \leq OPT \cdot (1 + 2\delta)$. By Lemma 15, the paths $P^{m_0,m_1}_{s_1,t_1}$ and $P^{m_0,m_1}_{s_2,t_2}$ found in Step 6 are non-conflicting and $len(P^{m_0,m_1}_{s_1,t_1}) + len(P^{m_0,m_1}_{s_2,t_2}) \leq \xi \leq (1 + 2\delta) \cdot OPT$.

If $\{m_0, m_1\}$ is an intersecting-pivot, then by definition, all the edges that P_{m_0,m_1} shares with P_{s_1,t_1} or P_{s_2,t_2} are edges common to both. Consequently, the paths found in Step 3 and 4 have overall at most $OPT - 2 \cdot len(P_{m_0,m_1})$ edges. Hence, $\xi \leq OPT$. In this case, by Lemma 15,the paths $P^{m_0,m_1}_{s_1,t_1}$ and $P^{m_0,m_1}_{s_2,t_2}$ found in Step 6 are non-conflicting and $len(P^{m_0,m_1}_{s_1,t_1}) + len(P^{m_0,m_1}_{s_2,t_2}) \leq \xi$. Consequently, since $\xi \leq OPT$, these paths are an optimal pair of paths. ■ We prove the following lemma in Appendix 6.

Lemma 15 *Let (G, s_1, t_1, s_2, t_2) be the input to Algorithm 1. Assume that Q, P'_{t_1,m_i}, $P'_{t_2,m_{1-i}}$, P'_{s_2,m_j}, $P'_{t_1,m_{1-j}}$ and P_{m_0,m_1} are as computed by Algorithm 1 in an iteration using $\{m_0, m_1\}$. Let $\xi = len(P'_{s_1,m_i}) + len(P'_{t_2,m_{1-i}}) + len(P'_{s_2,m_j}) + len(P'_{t_1,m_{1-j}}) + 2 \cdot len(P_{m_0,m_1})$. Then there exists a procedure that runs in time polynomial in n that finds non-conflicting paths $P^{m_0,m_1}_{s_1,t_1}$ and $P^{m_0,m_1}_{s_2,t_2}$ in Q with $len(P^{m_0,m_1}_{s_1,t_1}) + len(P^{m_0,m_1}_{s_2,t_2}) \leq \xi$.*

5 Algorithm 2

The input to Algorithm 2 consists of an instance (G, s_1, t_1, s_2, t_2), $\gamma > 0$ and $d \in [n]$. The additional parameter d is required for using this algorithm in both the additive and multiplicative approximation modes. We prove here that, if $\gamma \cdot OPT \leq \gamma d \leq \Delta \cdot OPT$, then Algorithm 2 returns an optimal pair of paths in time $(n/(\gamma d))^{O(1/\gamma)} \cdot poly(n)$.

Algorithm 2 is a variation of the basic algorithm described in Section 3, which works when the input instance has an optimal pair of paths that are far from each other. It is used because it has a better running time when OPT is large, which is essential for the additive approximation. We next explain how it differs from the basic algorithm.

Suppose that the input (G, s_1, t_1, s_2, t_2) satisfies $\gamma \cdot OPT \leq \gamma d \leq \Delta \cdot OPT$ and that P_{s_1,t_1} and P_{s_2,t_2} are an optimal pair of paths that are $\Delta \cdot OPT$ apart. Recall that the basic algorithm, required finding specific vertices $u_1, u_2, \ldots, u_h$ in P_{s_1,t_1}. These vertices can be found via exhaustive search over all relevant subsets of $V(G)$. Algorithm 2 is almost the same as the basic algorithm except that the vertices $u_1, u_2, \ldots, u_h$ are selected from a subset of $V(G)$, which we call *representatives*, and this subset may be significantly smaller than $V(G)$. The relevant parameters for choosing this set are γ and d.

A set of representatives S has the property that every vertex in $V(G)$ is very close to a vertex in S. Consequently, the approach used in the basic algorithm will work when we use representatives. We now formally define the set of representatives and prove that such a set always exists. We then present the algorithm and prove its correctness.

Definition 16 $[Rep_G(\ell)]$ *Given G and $\ell > 0$, let $Rep_G(\ell)$ be an arbitrary subset of $V(G)$ such that: (i) for every $u \in V(G)$, there exists $v \in Rep_G(\ell)$ such that $dist(u, v) < \ell$; and (ii) $|Rep_G(\ell)| \leq 2n/\ell$.*

Lemma 17 *For every connected graph G and $\ell > 0$, there exists a set $Rep_G(\ell)$ satisfying Definition 16.*

Proof. Initially set $Rep_G(\ell) = \{u\}$, where u is an arbitrary vertex from $V(G)$. Afterwards add vertices to $Rep_G(\ell)$ in the following manner. If there is a vertex in $V(G) \setminus Rep_G(\ell)$ whose distance from every other vertex in $Rep_G(\ell)$ is greater than ℓ, then add it to $Rep_G(\ell)$, otherwise stop. This process eventually ends since $V(G)$ is finite. Every vertex in $V(G)$ has distance not exceeding ℓ to some vertex in $Rep_G(\ell)$ because either it is in the set or it was not added. Thus, the minimum distance between any pair of distinct vertices in $Rep_G(\ell)$ is ℓ. Therefore, since G is connected, if $|Rep_G(\ell)| > 1$, then for all $v \in Rep_G(\ell)$ there are at least $\lceil \ell/2 \rceil$ distinct vertices (including v itself) whose distance from v is less than their distance to any other vertex in $Rep_G(\ell)$. Consequently, $|Rep_G(\ell)| \leq 2n/\ell$. ■

Algorithm 2

Input: an instance (G, s_1, t_1, s_2, t_2), $\gamma > 0$ and $d \in [n]$

1. $P^*_{s_1,t_1} \longleftarrow \emptyset$, $P^*_{s_2,t_2} \longleftarrow \emptyset$
2. Iterate over all $S^* \subseteq Rep_G(\gamma d/4)$ such that $|S^*| \leq \lceil 8/\gamma \rceil$
   (a) $P'_{s_1,t_1} \longleftarrow$ an arbitrary shortest path in $\left(V(G), \bigcup_{v \in S^* \cup \{s_1,t_1\}} B_G(v, \gamma d/2)\right)$ between s_1 and t_1, if one exists, and $\emptyset$ otherwise
   (b) $P'_{s_2,t_2} \longleftarrow$ an arbitrary shortest path in $(V, E(G) \setminus E(P'_{s_1,t_1}))$ path between s_2 and t_2, if one exists, and $\emptyset$ otherwise
   (c) If $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ are empty, then $P^*_{s_1,t_1} \longleftarrow P'_{s_1,t_1}$, $P^*_{s_2,t_2} \longleftarrow P'_{s_2,t_2}$
   (d) If P'_{s_1,t_1} and P'_{s_2,t_2} are both non-empty and $len(P'_{s_1,t_1}) + len(P'_{s_2,t_2}) < len(P^*_{s_1,t_1}) + len(P^*_{s_2,t_2})$, then $P^*_{s_1,t_1} \longleftarrow P'_{s_1,t_1}$, $P^*_{s_2,t_2} \longleftarrow P'_{s_2,t_2}$

Output: $P^*_{s_1,t_1}, P^*_{s_2,t_2}$

Theorem 18 *Let $\gamma > 0$ and $d \in [n]$. Assume that Algorithm 2 is executed with parameters $(G, s_1, t_1, s_2, t_2), \gamma$ and d. If (G, s_1, t_1, s_2, t_2) is disjoint and $\gamma \cdot OPT \leq \gamma d \leq \Delta \cdot OPT$, then Algorithm 2 will return an optimal pair of paths. The running time of Algorithm 2 is $(n/(\gamma d))^{O(1/\gamma)} \cdot poly(n)$.*

Proof. The running time follows, since the iteration in Step 2 is executed $O(n/(\gamma d))$ choose $O(1/\gamma)$ times. The other steps in the algorithm only increase the running time by a multiplicative factor that is polynomial in n.

Let P_{s_1,t_1}, P_{s_2,t_2} be an optimal pair such that $dist(P_{s_1,t_1}, P_{s_2,t_2}) \geq \Delta \cdot OPT \geq \gamma d$. Since $|V(P_{s_1,t_1})| \leq OPT \leq d$, by Lemma 17, there exists $U = Rep_{P_{s_1,t_1}}(\gamma d/4)$, where $|U| \leq \lceil 8/\gamma \rceil$. Consider $Rep_G(\gamma d/4)$ as selected in the execution of Algorithm 2. For each $u \in U$, by Definition 16, we can choose $q_u \in Rep_G(\gamma d/4)$ such that $dist_G(q_u, u) < \gamma d/4$. Let $Q = \{q_u \mid u \in U\}$, then clearly $|Q| \leq |U|$. Step 2 of Algorithm 2 checks every subset of $Rep_G(\gamma d/4)$ of size at most $\lceil 8/\gamma \rceil$. Hence $S^* = Q$ for some iteration of Step 2. Immediately after executing the iteration of Step 2 where $S^* = Q$, let P'_{s_2,t_2} and P'_{s_1,t_1} be the paths found in Steps 2a and 2b, respectively.

We observe that, by the definition of U, for every vertex $v \in V(P_{s_1,t_1})$ there exists a vertex $u \in U$ such that $dist(v, u) \leq \gamma d/4$. By the choice of Q, for every $u \in U$ there exists $q \in Q$ such that $dist(q_u, u) \leq \gamma d/4$. Consequently, by the triangle inequality, for every vertex $v \in V(P_{s_1,t_1})$ there exists a vertex $q_u \in Q$ such that $dist(v, q_u) < \gamma d/2$. Hence, since $Q = S^*$, $E(P_{s_1,t_1}) \subseteq \bigcup_{v \in S^* \cup \{s_1,t_1\}} B_G(v, \gamma d/2)$ and therefore $len(P'_{s_1,t_1}) \leq len(P_{s_1,t_1})$.

We also observe that by triangle inequality, P_{s_2,t_2} and $\bigcup_{v \in S^* \cup \{s_1,t_1\}} B_G(v, \gamma d/2)$ are edge-disjoint since $\gamma d \leq \Delta \cdot OPT$ and $dist(q_u, P_{s_1,t_1}) \leq \gamma d/4$, for every $q_u \in Q$. Thus, P'_{s_1,t_1} and P_{s_2,t_2} are edge-disjoint. It follows as in the proof for the basic algorithm, in Section 3 following Definition 10, that P'_{s_1,t_1} and P'_{s_2,t_2} are an optimal pair of paths. ■

6 Main Results

We start this section by proving the reduction from the min-sum 2-paths orientation problem to the min-sum 2 edge-disjoint paths problem. Afterwards we prove the additive approximation result and we conclude the section by proving the multiplicative approximation result.

Theorem 19 *If there exists an approximation algorithm for the min-sum* 2 *edge-disjoint paths problem with time complexity $T(n)$, then there exists an algorithm for the min-sum* 2*-paths orientation problem with time complexity $T(n) + poly(n)$ and the same quality of approximation.*

Proof. Given an instance (G, s_1, t_1, s_2, t_2), we solve the min-sum 2-paths orientation problem as follows: (i) execute Algorithm 1 with input (G, s_1, t_1, s_2, t_2); (ii) execute the approximation algorithm for the min-sum 2 edge-disjoint paths problem with input (G, s_1, t_1, s_2, t_2); and then (iii) return an arbitrary best solution.

If the input instance is intersecting then, by Theorem 14, Algorithm 1 returns an optimal pair of paths. If the input instance is not intersecting then, by Lemma 12, it is disjoint. So G has an optimal pair of edge-disjoint paths. Thus, the approximation algorithm for the min-sum 2 edge-disjoint paths returns the required pair of paths. ■

Theorem 20 *There exists an algorithm that given an instance (G, s_1, t_1, s_2, t_2) and $\alpha > 0$, returns non-conflicting paths P_{s_1,t_1} and P_{s_2,t_2} such that $len(P_{s_1,t_1}) + len(P_{s_2,t_2}) \leq OPT + 2\alpha n$, in time $(1/\alpha)^{\tilde{O}(1/\alpha)} \cdot poly(n)$.*

Proof. To obtain the required paths we perform the following steps: (i) execute Algorithm 1 with input (G, s_1, t_1, s_2, t_2); (ii) execute Algorithm 2 with input $(G, s_1, t_1, s_2, t_2), \alpha$ and n; and then (iii) return an arbitrary best solution.

The bound on the running time is immediate from Theorem 14 and Theorem 18. By Theorem 14, Algorithm 1 returns a pair of non-conflicting paths $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ whose sum of lengths does not exceed $(1 + 2\delta) \cdot OPT$. Note that if $\delta > 0$, then $OPT \leq n$. Thus, if $\delta \cdot OPT \leq \alpha n$, then

$(1+2\delta)\cdot OPT \leq OPT + 2\alpha n$ and hence the theorem holds when $\delta \cdot OPT \leq \alpha n$. Suppose that $\delta \cdot OPT > \alpha n$ and therefore, $\alpha \cdot OPT \leq \alpha n \leq \delta \cdot OPT \leq \Delta \cdot OPT$ and hence by Theorem 18, Algorithm 2 will return an optimal pair of paths. Consequently, the theorem holds in general. ∎

We prove the following theorem in Appendix 6.

Theorem 21 *There exists an algorithm that, given an instance* (G, s_1, t_1, s_2, t_2) *and* $\gamma > 0$, *returns non-conflicting paths* P_{s_1,t_1} *and* P_{s_2,t_2} *such that* $len(P_{s_1,t_1}) + len(P_{s_2,t_2}) \leq (1+2\gamma)\cdot OPT$, *in time* $(n/(\gamma \cdot OPT))^{O(1/\gamma)} \cdot poly(n)$.

References

[1] J. Han and F. Jahanian. Impact of path diversity on multi-homed and overlay networks. In *DSN*, page 29, 2004.

[2] R. Hassin and N. Megiddo. On orientations and shortest paths. In *Linear Algebra Appl.*, 1989.

[3] T. Ito, Y. Miyamoto, H. Ono, H. Tamaki, and R. Uehara. Route-enabling graph orientation problems. In *ISAAC*, pages 403–412, 2009.

[4] F. Kammer and T. Tholey. The k-disjoint paths problem on chordal graphs. In *WG*, pages 190–201, 2009.

[5] Y. Kobayashi and Ch. Sommer. On shortest disjoint paths in planar graphs. In *ISAAC*, pages 293–302, 2009.

[6] C. Li, T. S. McCormick, and D. Simich-Levi. The complexity of finding two disjoint paths with min-max objective function. *Discrete Appl. Math.*, 26(1):105–115, 1989.

[7] N. Robertson and P. D. Seymour. Graph minors. XIII. The disjoint paths problem. *J. Comb. Theory Ser. B*, 63(1):65–110, January 1995.

[8] V. Vasudevan, D.G. Andersen, and H. Zhang. Understanding the AS-level path disjointness provided by multi-homing. Technical report, 2007.

[9] B. Yang and S. Q. Zheng. Finding min-sum disjoint shortest paths from a single source to all pairs of destinations. In *TAMC*, pages 206–216, 2006.

[10] P. Zhang and W. Zhao. On the complexity and approximation of the min-sum and min-max disjoint paths problems. In *ESCAPE*, pages 70–81, 2007.

Appendix: Proof of Lemma 12

Lemma 12 *Let (G, s_1, t_1, s_2, t_2) be an instance for which $OPT < \infty$, then (G, s_1, t_1, s_2, t_2) is disjoint or intersecting (or both).*

Proof. If G is disjoint, then the lemma trivially holds and hence, we may assume that it is not. Let $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ be an optimal pair of paths in G. Obviously, these paths are not edge-disjoint. Let $Q = (V(P^*_{s_1,t_1}) \cup V(P^*_{s_2,t_2}), E(P^*_{s_1,t_1}) \cup E(P^*_{s_2,t_2}))$ be an undirected graph. We note that

$$|E(Q)| \leq OPT - |E(P^*_{s_1,t_1}) \cap E(P^*_{s_2,t_2})|. \tag{1}$$

Let $\{a_1, \ell_1\}$ and $\{a_2, \ell_2\}$ be edges in $E(P^*_{s_1,t_1}) \cap E(P^*_{s_2,t_2})$ such that $dist_{P^*_{s_1,t_1}}(s_1, a_1)$ and $dist_{P^*_{s_2,t_2}}(s_2, a_2)$, are minimum over all vertices a_1 and a_2 in $V(P^*_{s_1,t_1}) \cap V(P^*_{s_2,t_2})$. Let P_{s_1,a_1} be a subpath of $P^*_{s_1,t_1}$ and P_{s_2,a_2} be a subpath of $P^*_{s_2,t_2}$. For an illustration see Figure 2. We divide the proof into 4 cases.

(a) Suppose that $a_1 = a_2 = a$. Let P_{s_1,s_2} be the concatenation of P_{s_1,a_1} and P_{s_2,a_2}. Observe that there exists a path P_{t_1,t_2} in Q that is edge-disjoint from P_{s_1,s_2}. Let $P_{x,y}$ be a path in Q such that $len(P_{x,y}) = dist_Q(P_{s_1,s_2}, P_{t_1,t_2})$, x is in one of the paths P_{s_1,s_2} and P_{t_1,t_2} and y is in the other. We note that, $P_{x,y}$ does not share edges either with P_{s_1,s_2} or with P_{t_1,t_2}, since otherwise we get a contradiction to $len(P_{x,y}) = dist_Q(P_{s_1,s_2}, P_{t_1,t_2})$.

Let P_{s_1,t_1} be the concatenation of the subpath of P_{s_1,s_2} between s_1 and x, $P_{x,y}$ and the subpath of P_{t_1,t_2} between y and t_1. In the same manner let P_{s_2,t_2} be the concatenation of the subpath of P_{s_1,s_2} between s_2 and x, $P_{x,y}$ and the subpath of P_{t_1,t_2} between y and t_2. We note that both $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ have a subpath between a vertex in P_{s_1,s_2} and a vertex P_{t_1,t_2} and that both of these subpaths are not shorter than $P_{x,y}$. Hence that sum of length of P_{s_1,t_1} and P_{s_2,t_2} does not exceed that of $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$. Consequently, (G, s_1, t_1, s_2, t_2) is intersecting.

(b) Assume for the sake of contradiction that $a_1 \neq a_2$ and either $\ell_1 = a_2$ or $\ell_2 = a_1$. Since $dist_{P^*_{s_1,t_1}}(s_1, a_1) < dist_{P^*_{s_1,t_1}}(s_1, a_2)$, if $P^*_{s_1,t_1}$ is treated as a directed path the arc replacing $\{a_1, a_2\}$ is (a_1, a_2). By similar reasoning with $P^*_{s_2,t_2}$, the arc replacing $\{a_1, a_2\}$ is (a_2, a_1). Thus, a contradiction to $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ being non-conflicting.

Note that the this analysis also works when $s_1 = \ell_2$ or $s_2 = \ell_1$. Hence from here on we also assume that $s_1 \neq \ell_2$ and $s_2 \neq \ell_1$.

(c) Assume for the sake of contradiction that $a_1 \neq a_2$ and $\ell_1 = \ell_2 = \ell$. Since $dist_{P^*_{s_1,t_1}}(s_1, \ell) < dist_{P^*_{s_1,t_1}}(s_1, a_2)$, if P_{s_1,t_1} is treated as a directed path, then the arc replacing $\{\ell, a_2\}$ is (ℓ, a_2). By similar reasoning, if $P^*_{s_2,t_2}$ is treated as a directed path, then the arc replacing $\{\ell, a_2\}$ is (a_2, ℓ). Thus, a contradiction to $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ being non-conflicting.

(d) Assume for the sake of contradiction that $a_1 \neq a_2$ and $\ell_1 \neq \ell_2$. By similar reasoning to the above, if $P^*_{s_1,t_1}$ is treated as a directed path, then the arc replacing $\{a_1, \ell\}$ is (a_1, ℓ) and, if $P^*_{s_2,t_2}$ is treated as a directed path, then the arc replacing $\{a_2, \ell_2\}$ is (a_2, ℓ_2). Hence, according to the definition of a_1, the path $P^*_{s_1,t_1}$ has a subpath P_{ℓ_1,a_2} which does not contain ℓ_2 and, when treated as a directed path its edges are directed towards a_2. In the same manner, there exists a subpath

P_{ℓ_2,a_1} of $P^*_{s_2,t_2}$ which does not contain ℓ_1 and when treated as a directed path its edges are directed towards a_1. For an illustration see Figure 2.

Finally, let P_{ℓ_2,t_1} be a subpath of $P^*_{s_1,t_1}$ and P_{ℓ_1,t_2} a subpath of $P^*_{s_2,t_2}$. Note that by construction P_{s_1,a_1}, P_{ℓ_1,a_2} and P_{ℓ_2,t_1} are pairwise-edge-disjoint, as are P_{s_2,a_2}, P_{ℓ_2,a_1} and P_{ℓ_1,t_2}.

Assume that P_{ℓ_2,t_1} and P_{ℓ_2,a_1} are edge-disjoint, and so are P_{ℓ_1,t_2} and P_{ℓ_1,a_2}.

Let P_{s_1,t_1} be the concatenation of P_{s_1,a_1}, and P_{a_1,ℓ_2} and P_{ℓ_2,t_1}. Observe that P_{s_1,t_1} is a path since P_{s_1,a_1} and P_{a_1,ℓ_2} are edge-disjoint by the definition of a_1. In the same manner, let P_{s_2,t_2} be the concatenation of P_{s_2,a_2}, and P_{a_2,ℓ_1} and P_{ℓ_1,t_2}. By the same reasoning as for P_{s_1,t_1}, we may conclude that P_{s_2,t_2} is also a path.

As the edges $\{a_1, \ell_1\}$ and $\{a_2, \ell_2\}$ are not in $E(P_{s_1,t_1})$ or $E(P_{s_2,t_2})$, we have $len(P_{s_1,t_1}) + len(P_{s_2,t_2}) < len(P^*_{s_1,t_1}) + len(P^*_{s_2,t_2}) = OPT$. Note that P_{s_1,t_1} and P_{s_2,t_2} are non-conflicting since the only problem that may arise is between P_{ℓ_2,t_1} and P_{ℓ_2,a_1}, and P_{ℓ_1,t_2} and P_{ℓ_1,a_2}. Yet, by assumption, these pairs of paths do not intersect. Thus, P_{s_1,t_1} and P_{s_2,t_2} are non-conflicting and $len(P_{s_1,t_1}) + len(P_{s_2,t_2}) < OPT$, in contradiction to Observation 8.

Now assume that P_{ℓ_2,t_1} and P_{ℓ_2,a_1} are not edge-disjoint. The only problem this may cause is that P_{s_1,t_1} as defined for the edge-disjoint case is not a path. This can be resolved by simply removing any cycles from P_{s_1,t_1}. The same holds if instead or in addition P_{ℓ_1,t_2} and P_{ℓ_1,a_2} are not edge-disjoint. Now the proof proceeds in the same manner as the previous case.

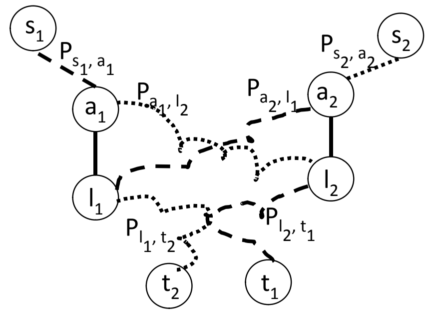


Figure 2: $a_1 \neq a_2$, and $\ell_1 \neq \ell_2$, and P_{ℓ_2,t_1} and P_{ℓ_2,a_1} are edge-disjoint and so are P_{ℓ_1,t_2} and P_{ℓ_1,a_2}.

∎

Appendix: Proof of Lemma 15

Lemma 15 *Let (G, s_1, t_1, s_2, t_2) be the input to Algorithm 1. Assume that Q, P'_{t_1,m_i}, $P'_{t_2,m_{1-i}}$, P'_{s_2,m_j}, $P'_{t_1,m_{1-j}}$ and P_{m_0,m_1} are as computed by Algorithm 1 in an iteration using $\{m_0, m_1\}$. Let $\xi = len(P'_{s_1,m_i}) + len(P'_{t_2,m_{1-i}}) + len(P'_{s_2,m_j}) + len(P'_{t_1,m_{1-j}}) + 2 \cdot len(P_{m_0,m_1})$. Then there exists a procedure that runs in time polynomial in n that finds non-conflicting paths $P^{m_0,m_1}_{s_1,t_1}$ and $P^{m_0,m_1}_{s_2,t_2}$ in Q with $len(P^{m_0,m_1}_{s_1,t_1}) + len(P^{m_0,m_1}_{s_2,t_2}) \leq \xi$.*

Proof. Without loss of generality we may assume that $i = 1$.

We use Figures 3, 4 and 5 to illustrate various cases in the proof. The solid lines in these figures correspond to edge-disjoint paths. The paths P'_{s_1,m_1}, P'_{t_2,m_0} are shown in all of the figures, but in general only parts of P'_{s_2,m_j} and $P'_{t_1,m_{1-j}}$ are shown. For example, in Figure 3b, the path between f_2 and m_1 is a subpath of P'_{s_1,m_1}, but it is not necessarily a subpath of P'_{s_2,m_j}. The arrowed dotted lines in the figures represent the non-conflicting paths we find. These paths are not fully shown in Figures 4 and 5, instead in all these figures there is a cycle represented by thick lines. The non-conflicting paths go through the cycle in the same direction which may be either clockwise or counter-clockwise. Part of our proof is to show that at least one of the these options gives the required non-conflicting paths.

Let P_{s_1,t_2} be the concatenation of the paths P'_{s_1,m_1}, P_{m_0,m_1} and P'_{t_2,m_0}. A path $P_{u,v}$ in Q is a *hop* if it is between a vertex in P'_{s_1,m_1} and a vertex in P'_{t_2,m_2} and is edge-disjoint with both of these paths. We note that P_{m_0,m_1} is a hop.

Let f_1 be the first vertex on $P'_{t_1,m_{1-j}}$ that is also on P_{s_1,t_2}. In the same manner, let f_2 be the first vertex on P'_{s_2,m_j} that is also on P_{s_1,t_2}. For an illustration see Figure 3a. Let l_1 be the first vertex on $P'_{t_1,m_{1-j}}$ that is on a hop P_{l_1,a_1} and let l_2 be the first vertex on P'_{s_2,m_j} that is on a hop P_{l_2,a_2}. We note that the hop P_{l_1,a_1} is a subpath of $P'_{t_1,m_{1-j}}$ and the hop P_{l_2,a_2} is a subpath of P'_{s_2,m_j}.

An explicit depiction of these paths can be found in Figure 4. We note that both P_{l_2,a_2} are P_{l_1,a_1} are not shown in all figures and that, in Figure 3b, $P_{l_2,a_2} = P_{l_1,a_1} = P_{m_1,m_0}$.

Observation 22 *$dist_{P_{s_1,t_2}}(f_1, l_1) \leq dist_{P'_{t_1,m_{1-j}}}(f_1, l_1)$ and $dist_{P_{s_1,t_2}}(f_2, l_2) \leq dist_{P'_{s_2,m_j}}(f_2, l_2)$.*

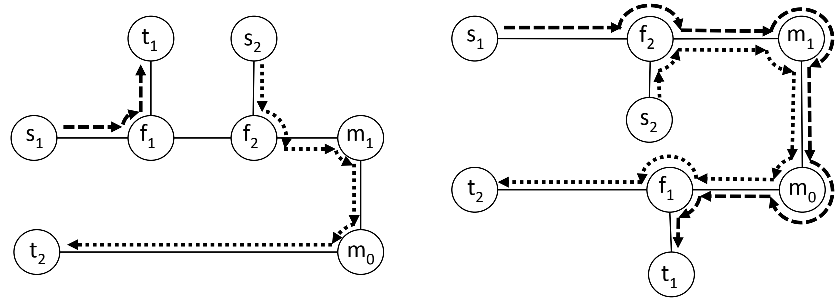
Proof. Assume for the sake of contradiction that $dist_{P'_{t_1,m_{1-j}}}(f_1, l_1) < dist_{P_{s_1,t_2}}(f_1, l_1)$, then either P'_{s_1,m_1} or P'_{t_2,m_0} can be replaced by a shorter path by using a shorter path between f_1 and l_1 which contradicts the choice of P'_{s_1,m_1} and P'_{t_2,m_0}. The second inequality follows similarly. ■

We now analyze the different cases. If $dist_{P_{s_1,t_2}}(s_1, f_1) \leq dist_{P_{s_1,t_2}}(s_1, f_2)$, then we can take the pair of paths depicted in Figure 3a. Clearly this pair of paths satisfies the requirement of the lemma. So, from here on we assume that $dist_{P_{s_1,t_2}}(s_1, f_1) > dist_{P_{s_1,t_2}}(s_1, f_2)$.

Suppose that P_{m_1,m_2} is the only hop. Then using the previous inequality and the fact that P'_{s_2,m_j} and $P'_{t_1,m_{1-j}}$ are both edge disjoint from P_{m_1,m_2}, it follows that f_1 is on P'_{t_2,m_0} and f_2 is on P'_{s_1,m_1}. This case is shown in Figure 3b. We note that in this case $l_1 = m_0$ and $l_2 = m_1$. By Observation 22, the pair of paths depicted in Figure 3b satisfies the requirement of the lemma. The same reasoning holds for the case depicted in Figure 4a.

In the cases shown in Figures 4b and 5, as mentioned before, we give two options for filling in the missing parts of the paths, viz. clockwise and counter-clockwise. By Observation 22, if we sum up the length of the paths in the two options we get at most 2ξ and hence at least one of these options satisfies the requirements of the lemma.

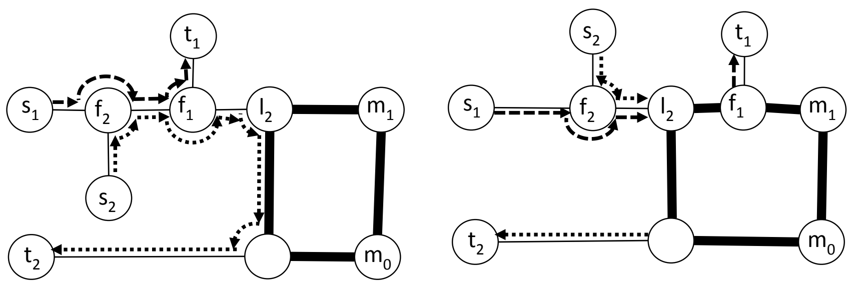
We note that the same analysis as for the cases with the cycle also holds when s_1, t_1, s_2 and t_2 are connected to a cycle via edge-disjoint paths. Finally, all other cases are similar to one of the cases depicted in Figures 4 and 5. That is, to get one of the covered cases, exchange the roles of s_1 and t_1 and the roles of s_2 and t_2. Notice that when f_1 is on a cycle represented by thick lines, as in Figure 5b, the analysis is independent of its exact location. This also holds for f_2.



(a) $dist_{P_{s_1,t_2}}(s_1, f_1) \leq dist_{P_{s_1,t_2}}(s_1, f_2)$

(b) Single hop

Figure 3:

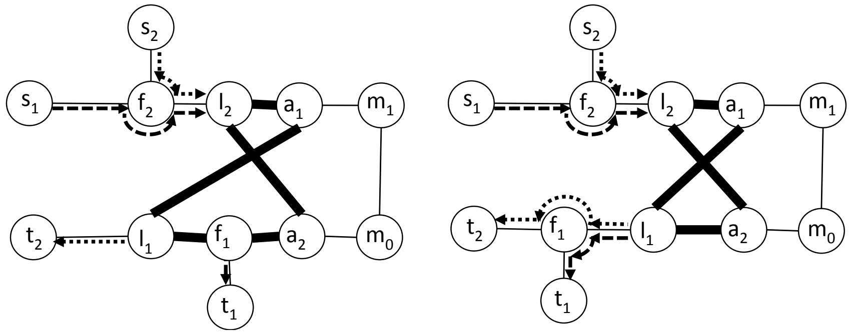


(a) Cycle 1

(b) Cycle 2

Figure 4:

■



(a) Cycle 3 (b) Cycle 4

Figure 5:

Appendix: Proof of Theorem 21

Theorem 21 *There exists an algorithm that, given an instance* (G, s_1, t_1, s_2, t_2) *and* $\gamma > 0$*, returns non-conflicting paths* P_{s_1,t_1} *and* P_{s_2,t_2} *such that* $len(P_{s_1,t_1}) + len(P_{s_2,t_2}) \leq (1 + 2\gamma) \cdot OPT$*, in time* $(n/(\gamma \cdot OPT))^{O(1/\gamma)} \cdot poly(n)$*.*

Proof. To obtain the required paths we perform the following steps.

1. Compute shortest paths P'_{s_1,t_1} between s_1 and t_1, and P'_{s_2,t_2} between s_2 and t_2. If they are non-conflicting, then return these and stop.

2. Execute Algorithm 1 with input G, s_1, t_1, s_2 and t_2, and let x be the sum of lengths of the paths returned.

3. For each $d \in \{\lfloor \frac{x}{2} \rfloor, \ldots, x\}$ execute Algorithm 2 with input $(G, s_1, t_1, s_2, t_2), d$ and γ.

4. Return an arbitrary best solution.

The bound on the running time is immediate from Theorem 14 and Theorem 18.

If we stopped at Step 1, then P'_{s_1,t_1} and P'_{s_2,t_2} are an optimal pair of paths. Hence, we may assume that we did not stop at Step 1. Consequently, P'_{s_1,t_1} and P'_{s_2,t_2} are not edge-disjoint. Therefore, at least one of $dist_G(s_1, s_2), dist_G(s_1, t_2), dist_G(s_2, t_1)$ and $dist_G(t_1, t_2)$ is less than $OPT/2$. Thus, by the definition of δ, we may assume that $\delta \leq 1/2$.

By Theorem 14, Algorithm 1 returns a pair of non-conflicting paths $P^*_{s_1,t_1}$ and $P^*_{s_2,t_2}$ whose sum of lengths does not exceed $(1+2\delta) \cdot OPT$. Thus, if $\delta \leq \gamma$, then this does not exceed $(1+2\gamma) \cdot OPT$ and hence the theorem holds when $\delta \leq \gamma$. Suppose that $\gamma < \delta$. Since $\delta \leq 1/2$, by the above, $OPT \leq x \leq 2 \cdot OPT$. Thus $OPT \in \{\lfloor x/2 \rfloor, \ldots, x\}$ and hence, at some stage in the execution of Step 3, Algorithm 2 was called with parameter $d = OPT$. In this case we have $\gamma \cdot OPT = \gamma \cdot d < \delta \cdot OPT < \Delta \cdot OPT$. Consequently, by Theorem 18, Algorithm 2 returned an optimal pair of paths. Thus, the theorem holds in general. ■