Computing and Informatics, Vol. 32, 2013, 23–45

COMPLEXITY AND APPROXIMATION RESULTS FOR THE MIN-SUM AND MIN-MAX DISJOINT PATHS PROBLEMS

Peng Zhang

School of Computer Science and Technology
Shandong University
Jinan 250101, China
e-mail: algzhang@sdu.edu.cn

Wenbo Zhao

Department of Computer Science and Engineering
University of California, San Diego
La Jolla, CA 92093, USA
e-mail: w3zhao@ucsd.edu

Daming Zhu

School of Computer Science and Technology
Shandong University
Jinan 250101, China
e-mail: dmzhu@sdu.edu.cn

Communicated by Marian Vojteršic

Abstract. Given a graph $G = (V, E)$ and k source-sink pairs $\{(s_1, t_1), \ldots, (s_k, t_k)\}$ with each $s_i, t_i \in V$, the Min-Sum Disjoint Paths problem asks to find k disjoint paths connecting all the source-sink pairs with minimized total length, while the Min-Max Disjoint Paths problem asks for k disjoint paths connecting all the source-sink pairs with minimized length of the longest path. We show that the weighted Min-Sum Disjoint Paths problem is $\mathrm{FP}^{\mathrm{NP}}$-complete in general graphs, and the unweighted Min-Sum Disjoint Paths problem and the unweighted Min-Max Disjoint

Paths problem cannot be approximated within $\Omega(m^{1-\epsilon})$ for any constant $\epsilon > 0$ even in planar graphs, assuming P $\neq$ NP, where m is the number of edges in G. We give for the first time a simple bicriteria approximation algorithm for the unweighted Min-Max Edge-Disjoint Paths problem and the weighted Min-Sum Edge-Disjoint Paths problem, with guaranteed approximation ratio $O(\log k/\log\log k)$ and $O(1)$, respectively.

Keywords: Disjoint paths, min-sum, min-max, computational complexity, approximation algorithms

Mathematics Subject Classification 2010: 68Q17, 68Q25, 68W25

1 INTRODUCTION

The disjoint paths problem is a classical problem in combinatorial optimization and graph theory. The problem finds its many applications in practical areas such as network routing and VLSI-design. Nowadays the disjoint paths problem has attracted much attention due to the rapid development of the communication network technology. See the monograph due to Korte et al. [11] and the remarkable thesis of Kleinberg [8] for more details. The Maximum Vertex-Disjoint Paths problem (MVDP) and the Maximum Edge-Disjoint Paths problem (MEDP) are two classical disjoint paths problems, which ask to find vertex- (edge-) disjoint paths to connect as many as possible source-sink pairs in a given graph [8, 2, 9]. In this paper, we study the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem.

1.1 The Problems and the Notation

Two paths are said to be *vertex-disjoint* if they do not share any vertex, and are said to be *edge-disjoint* if they do not share any edge. In the Min-Sum Disjoint Paths problem, we are given a graph $G = (V, E)$ with weight (a.k.a. length) $w_e \geq 0$ defined for every $e \in E$, and a set of k source-sink vertex pairs $D = \{(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)\}$. The goal of the problem is to find k disjoint paths to connect every source-sink pair, such that the total length of these paths is minimized. The instance of the Min-Max Disjoint Paths problem is the same as that of Min-Sum Disjoint Paths, while the problem is to find k disjoint paths to connect every source-sink pair with minimized length of the longest path.

If the weights are identical for all edges (without loss of generality, we may assume that $w_e = 1$ for every edge e), then we call the problem *unweighted*. According to the convention, the terms EDP and VDP denote the decision problems of MEDP and MVDP, respectively. When we talk about disjoint paths, we mean edge-disjoint paths and/or vertex-disjoin paths. Source and sink are also called *terminals*. We

interchangeably use the term source-sink pair and the term terminal pair. We say that an instance of the problem is *feasible* if all source-sink pairs can be connected by disjoint paths.

Throughout this paper, we use n and m to denote, respectively, the number of vertices and the number of edges in graph G. As usual, we use I to denote an instance of some problem. Assume I is an instance of some minimization problem Π. Let $\mathrm{OPT}(I)$ denote the cost of an optimal solution to I, and $A(I)$ denote the cost of an approximate solution found by algorithm $\mathcal{A}$ on instance I, where $\mathcal{A}$ is an approximation algorithm dealing with the problem Π. The approximation ratio of algorithm $\mathcal{A}$ is defined as the supremum[1] of $\frac{A(I)}{\mathrm{OPT}}(I)$ over all instances of problem Π. As we are dealing with the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem, we also use $\mathrm{OPT}_{\mathrm{MS}}(I)$ and $\mathrm{OPT}_{\mathrm{MM}}(I)$ for clarity to denote the cost of optimal solutions to the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem, respectively. For example, for the Min-Max EDP problem and its approximation algorithm $\mathcal{A}$, $A(I)$ should be the length of the longest path in the solution found by algorithm $\mathcal{A}$ on instance I, while $\mathrm{OPT}_{\mathrm{MM}}(I)$ should be the length of the longest path in an optimal solution to instance I.

1.2 Related Works

In general, the disjoint paths problem appears to be a hard problem. Karp [7] proved the classical undirected VDP and EDP problems (given a graph and k terminal pairs, the problem asks whether there are k vertex- (edge-) disjoint paths to connect all the terminal pairs) are NP-complete. For directed graphs, Fortune, Hopcroft and Wyllie proved that the VDP and EDP problems are NP-complete even for $k = 2$ (i.e., there are only two terminal pairs). The disjoint paths problem becomes tractable only for several classes of graphs such as undirected graphs with bounded tree-width [17] and directed acyclic graphs [4]. As there are many works about the disjoint paths problem, we mainly focus on its min-sum version and min-max version in the following.

First we state some works about the Min-Sum Disjoint Paths problem. Suurballe and Tarjan [18] gave a polynomial-time algorithm for finding a pair of edge-disjoint paths from one specified source to each possible sink of shortest total length in a directed graph. Yang and Zheng [21] extended the algorithm in [18] to solve the problem that finds the shortest two disjoint paths from one source s to a pair of sinks (t_1, t_2) for every such pair. This problem is equivalent to the Min-Sum EDP problem with constraints that $k = 2$ and $s_1 = s_2$.

For the general case that there are k terminal pairs in the problem, Brandes, Neyer and Wagner [1] proved that the unweighted Min-Sum EDP problem and the unweighted Min-Max EDP problem are NP-hard in planar graphs, even if the graph fulfills the Eulerian condition and the maximum degree of the graph is 4.

[1] In general, to calculate precisely the supremum is not easy. People often use a good upper bound as the approximation ratio.

For planar graphs, some cases of the min-sum disjoint paths problem are solvable in polynomial time. Colin de Verdière and Schrijver [3] showed that the Min-Sum VDP problem can be solved in $O(kn \log n)$ time for a directed or undirected planar graph G, provided that $s_1, s_2, \cdots, s_k$ are incident with a face s, and $t_1, t_2, \cdots, t_k$ are incident with a face t, where s and t are two distinct faces of G. By extending the results of [3], Kobayashi and Sommer [10] proved that the Min-Sum Disjoint Paths problem can be solved in polynomial time for planar graphs, when $k = 2$ and the terminals are incident with *at most* two faces.

Next we show the related works about the Min-Max Disjoint Paths problem. Li, Thomas and Simchi-Levi [12] considered the problem of finding two disjoint s-t paths such that the length of the longer path is minimized. They showed that both of the four versions of the problem, i.e., the graph may be directed or undirected, and the paths may be edge-disjoint or vertex-disjoint, are strongly NP-hard. They also gave a pseudo-polynomial time algorithm for the problem in directed acyclic graph. Kobayashi and Sommer [10] showed that the Min-Max Disjoint Paths problem when $k = 2$ can be solved in $O(n^2)$ time for undirected graphs with tree-width at most 2.

For the general case that there are k terminal pairs, the unweighted Min-Max EDP problem is NP-hard in planar graphs, even if the graph fulfills the Eulerian condition and the maximum degree is 4. This was proved by Brandes, Neyer and Wagner [1].

The bounded-length disjoint paths problem closely relates to the Min-Max Disjoint Paths problem. Itai, Perl and Shiloach [6] proved that it is NP-hard to find the maximum number of disjoint s-t paths such that every path has bounded length. Tragoudas and Varol [19] gave a stronger hardness result by showing that it is NP-complete to decide whether an unweighted graph contains a pair of edge disjoint s-t paths such that neither has more edges than a given length bound. They also gave a polynomial time algorithm to compute the maximum number of edge disjoint shortest s-t paths in a weighted graph.

1.3 Our Results

We systematically study the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem. There are several versions of the problems, according to

1. whether the input graph is directed or undirected,
2. whether the input graph is weighted or unweighted, and
3. whether the output paths are edge-disjoint or vertex-disjoint.

We prove new and strong complexity and approximation hardness results for the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem, generalizing the results of Brandes, Neyer and Wagner [1]. We give a simple bicriteria approximation algorithm for the first time to the unweighted Min-Max EDP problem and the weighted Min-Sum EDP problem, although the problems are very hard with respect to the approximation hardness as we prove.

Specifically, we show that the weighted Min-Sum Disjoint Paths problem is $\mathrm{FP}^{\mathrm{NP}}$-complete both in undirected and directed graphs, and the unweighted Min-Sum Disjoint Paths problem and the unweighted Min-Max Disjoint Paths problem cannot be approximated within $\Omega(m^{1-\epsilon})$ for any constant $\epsilon > 0$ even in planar graphs, unless P = NP. Notice that the approximation hardness of the unweighted problems implies the same hardness for the weighted problems. It is well known that the classical decision problems EDP and VDP (i.e., to decide whether the input instance is feasible) are NP-complete [7]. Although the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem include EDP and VDP as their subproblems, our results are strong since we prove all of the above complexity and approximation hardness results even if we know in advance that the given instance is feasible for EDP or VDP.

Then we give a simple bicriteria approximation algorithm for the unweighted Min-Max EDP problem and the weighted Min-Sum EDP problem, with bi-factors $(O(\log k/\log\log k), O(\log n/\log\log n))$ and $(O(1), O(\log n/\log\log n))$, respectively. The first factor in the bi-factor is the approximation ratio, while the second factor is the so-called *congestion*, that is, the maximum number of paths per edge in the solution. Our algorithm is based on randomized rounding and runs in polynomial time. In fact, our algorithm shows that whenever the linear program of the instance (for both of the two problems) has a fractional feasible solution, the instance admits an integral solution whose congestion is at most $O(\log n/\log\log n)$. Since solving linear program tells us whether the fractional feasible solution exists, our algorithm avoids to decide whether the instance is feasible for EDP, which is a NP-complete problem as previously mentioned.

2 COMPUTATIONAL COMPLEXITY OF WEIGHTED MIN-SUM DISJOINT PATHS

Recall that a feasible instance of the disjoint path problem means that all the source-sink pairs in the instance can be connected by (vertex- or edge-) disjoint paths. Given a graph, a set of k terminal pairs and a bound B, in the decision version of the Min-Sum VDP problem we are asked whether there are k vertex-disjoint paths connecting all the terminal pairs such that their total length is at most B. We can prove that the decision version of the Min-Sum VDP problem is NP-complete even if the problem is restricted in feasible (for VDP) instances, by a reduction from the VDP problem as follows. (The same holds for the decision version of the Min-Sum EDP problem. Here we just take the Min-Sum VDP problem as an example.) It is easy to see that the decision version of Min-Sum VDP is in NP. Let $I_1 = (G, D)$ be an instance of VDP. In the instance $I_2 = (G', D', B)$ of the decision version of Min-Sum VDP, let $D' = D$, $B = |E(G)|$, and G' be the graph G with one additional edge (s_i, t_i) for every terminal pair (s_i, t_i) whose weight is $B + 1$. Then I_1 is feasible iff there are k vertex-disjoint paths in I_2 whose total length is at most B.

In the following we shall prove a stronger complexity result, that is, the weighted Min-Sum Disjoint Paths problem in general graphs is $\mathrm{FP}^{\mathrm{NP}}$-compete. Recall that the complexity class $\mathrm{FP}^{\mathrm{NP}}$ is the set of all functions from strings to strings that can be computed in polynomial time by a deterministic Turing machine with access to a SAT oracle. $\mathrm{FP}^{\mathrm{NP}}$ is just the function version of the class P^{NP} (a.k.a. $\Delta_2\mathrm{P}$), which is at the second level of the polynomial hierarchy. The complexity class FNP, which is the function version of the class NP, is contained in $\mathrm{FP}^{\mathrm{NP}}$. If an optimization problem is $\mathrm{FP}^{\mathrm{NP}}$-complete, then the problem is NP-hard. But the opposite direction is believed not true. So in this sense $\mathrm{FP}^{\mathrm{NP}}$-complete is a stronger complexity conception than NP-hard.

It is known that the Max Weight SAT problem is $\mathrm{FP}^{\mathrm{NP}}$-complete [15]. In Max Weight SAT, we are given a formula ϕ in conjunctive normal form with positive integer weight for every clause. The problem is to find a truth assignment τ for the variables in ϕ such that the total weight of satisfied clauses under τ is maximized. We shall reduce Max Weight SAT to Min-Sum Disjoint Paths by the reduction between function problems (please refer to [15] for the definition) to prove that weighted Min-Sum Disjoint Paths is $\mathrm{FP}^{\mathrm{NP}}$-complete. We mention here that the well-known TSP problem is another important $\mathrm{FP}^{\mathrm{NP}}$-complete problem.

Lemma 1. Min-Sum Disjoint Paths is in $\mathrm{FP}^{\mathrm{NP}}$.

Proof. It is easy to see that the decision version of Min-Sum Disjoint Paths is in NP. Fix an instance of the Min-Sum Disjoint Paths problem. By scaling technique, we may assume that all the edge weights are integral. Let $w_{\min}$ be the minimum of edge weights, and W_{tot} be the total weight of edges in the instance. By a binary search in the interval $[k \cdot w_{\min}, W_{\mathrm{tot}}]$, using a SAT oracle, we can compute the minimum total length of such k vertex-disjoint (or edge-disjoint) paths. At each step we ask the oracle to decide an instance of the decision version of the Min-Sum Disjoint Paths problem. The binary search terminates when the width of the current search interval is less than 1. This requires $O(\log W_{\mathrm{tot}})$ steps, which is polynomial in the input size of the instance.

Let $\mathrm{OPT}_{\mathrm{MS}}$ be the total length of an optimal solution to the instance we just computed. Then, for every edge e, remove it temporarily from the graph, and make a query for the resulted decision problem with bound $B = \mathrm{OPT}_{\mathrm{MS}}$. If the answer is "yes", remove the edge from the graph permanently; otherwise put the edge back to the graph. By repeating the above procedure, we can eventually find all the edges that form an optimal solution to the problem. The lemma follows. □

Construction of the instance of Min-Sum EDP. First consider the Min-Sum EDP problem. Given an instance $I_1 = (\phi, w)$ of Max Weight SAT, we depict how to construct the instance $I_2 = (G, w, D)$ of Min-Sum EDP. Without loss of generality, we assume that each clause in ϕ contains at least 2 literals. We also assume that in every clause of ϕ, no literal appears more than once or appears together with its complement. We assign a gadget similar to that in Figure 1 to each variable x_i.

Inspired by the work due to Donald Knuth (which was cited in [7]), we get our gadget. As an example, assume variable x_1 appears positively in clauses c_{i_1} and c_{i_2}, and appears negatively in clauses c_{j_1}, c_{j_2} and c_{j_3}. Then we assign to x_1 the gadget in Figure 1. For each clause c_q (q should be one of i_1, i_2, j_1, j_2 and j_3), there is a terminal pair (s_q, t_q). For each terminal pair (s_q, t_q) there is a zigzag path connecting s_q and t_q and a thick edge also connecting s_q and t_q, called the *survival edge*. The survival edge for (s_q, t_q) has weight w_q (recall that w_q is the weight of the corresponding clause c_q in instance I_1), while all the non-survival edges have weight zero. In all these variable gadgets, the sources and sinks corresponding to the same clause are identical. Let G' be the resulted graph. Notice that in graph G' there is only one survival edge for each source-sink pair.

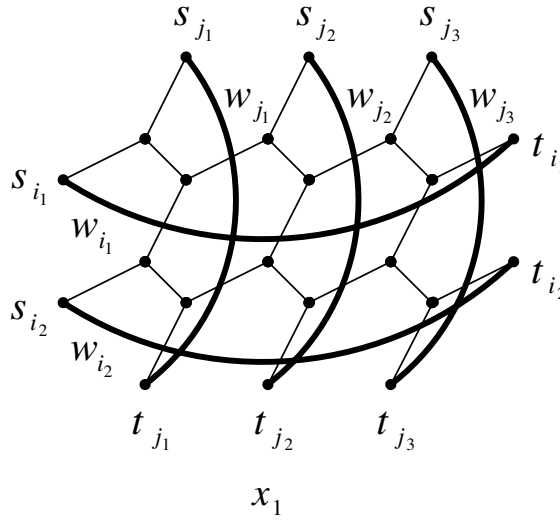

Fig. 1. Gadget for variable

Notice that in G' every terminal has degree strictly greater than 2. Let v be any terminal. Applying on v the transformation in Figure 2 [14] can decrease the degree of v by 1. By recursively applying on v the transformation, we can decrease the degree of v eventually to 2. All the newly introduced edges in the transformations have weight zero. The dashed edge (not belonging to the graph) in Figure 2, called the *demand edge*, indicates that its two endpoints form a source-sink pair. The transformation preserves the following invariant: if all the source-sink pairs introduced in the transformation are satisfied, then any path through v before the transformation must still pass through v after the transformation, and only one such path is allowed.

Now the degree of terminal v is 2 after the final transformation. This implies that the path satisfying other source-sink pair cannot pass through v, otherwise the source-sink pair of terminal v cannot be satisfied by disjoint paths. So for every

source-sink pair corresponding to some clause, the path connecting it is restricted within just one variable gadget. The resulting graph after the consecutive transformations is just our final graph G. This gives the instance I_2 of weighted Min-Sum EDP.

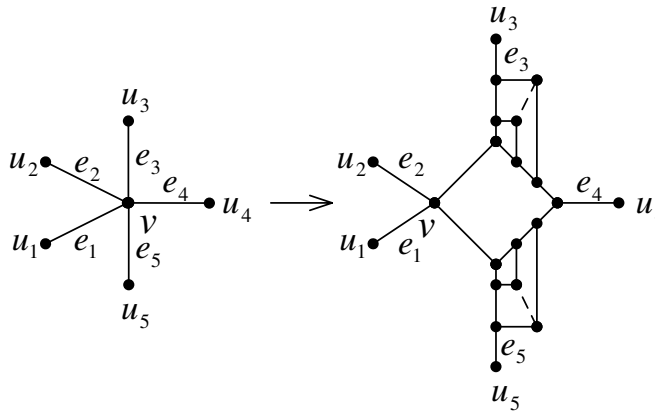

Fig. 2. The gadget used to decrease the degree of vertex v by one

Lemma 2. The reduction from the instance I_1 of Max Weight SAT to the instance I_2 of Min-Sum EDP can be finished in polynomial time.

Proof. For every variable, we can assume that the number of its occurrences in ϕ is bounded by the number of clauses in ϕ. Therefore the graph G' can be constructed in polynomial time. In graph G', a terminal v has degree being the number of literals in its corresponding clause. So the transformations used to decrease the degrees of terminals can also be finished in polynomial time. □

Denote by W_{tot} the total weight of all clauses in ϕ. Then we have Lemma 3.

Lemma 3. $\text{OPT}(I_1) = W_{\text{tot}} - \text{OPT}_{\text{MS}}(I_2)$.

Proof. Consider an optimal solution S^* of the instance I_2 with solution value $\text{OPT}_{\text{MS}}(I_2)$. For variable x_i, if S^* uses the horizontal zigzag paths in its corresponding gadget, then x_i is assigned the truth value 1. If S^* uses the vertical zigzag paths in its corresponding gadget, then x_i is assigned the truth value 0. Otherwise S^* does not use any zigzag path in the gadget of x_i, and in this case x_i is assigned any truth value. This gives a truth assignment τ for formula ϕ.

Then consider terminal pair (s_j, t_j) corresponding to clause c_j. If (s_j, t_j) is connected by zigzag path in the gadget of some variable x_i contained in c_j, then (s_j, t_j) consumes zero weight in the optimal solution S^*. Without loss of generality, suppose that (s_j, t_j) is connected by the horizontal zigzag path. By the construction of the gadget, variable x_i appears positively in clause c_j. Since x_i is assigned 1, c_j is

satisfied under τ. Conversely, if (s_j, t_j) is not connected by any zigzag path in the gadgets of variables in c_j, then (s_j, t_j) must be connected by the survival edge, which weighes w_j. So, in the gadget corresponding to each variable, say x_i, in c_j, there must be terminal pairs other than (s_j, t_j) that use the zigzag paths whose types (horizontal or vertical) are opposite to that of the zigzag path of (s_j, t_j) (according to our assumption, this zigzag path is not used by (s_j, t_j)). This implies that x_i is assigned a truth value such that its literal in c_j is false. Thus we know that all the literals in c_j are false, and hence c_j is unsatisfied under τ.

So, the total weight of satisfied clauses of formula ϕ under truth assignment τ is $W_{\text{tot}} - \text{OPT}_{\text{MS}}(I_2)$. Furthermore, τ must be an optimal solution to I_1, otherwise S^* is not optimal. The lemma follows. □

Theorem 1. Even restricted in feasible instances, the weighted Min-Sum EDP problem and the weighted Min-Sum VDP problem are FP^{NP}-complete both in undirected and directed graphs.

Proof. Since there are survival edges in the instance I_2 of Min-Sum EDP, I_2 is always feasible. Moreover, since all terminals have degree 2 in graph G, and all other vertices have degree 3, the edge-disjoint paths are identical to the vertex-disjoint paths in graph G. By Lemma 1, Lemma 2 and Lemma 3, the theorem follows in the case of undirected graphs.

For the directed case, it is easy to assign a direction for every edge in the variable gadget. Since for a source-sink pair the source has in-degree 0 and the sink has out-degree 0, we do not need the transformation in Figure 2. Again, in the resulting graph the edge-disjoint paths are identical to the vertex-disjoint paths, and we also have $\text{OPT}(I_1) = W_{\text{tot}} - \text{OPT}(I_2)$. We have finished the proof of the theorem. □

3 APPROXIMATION HARDNESS FOR MIN-SUM DISJOINT PATHS AND MIN-MAX DISJOINT PATHS

We first prove the approximation hardness for the unweighted Min-Sum Disjoint Paths problem in planar graphs. The hardness result can be directly extended to the unweighted Min-Max Disjoint Paths problem. Our proof is based on the work in [14] about the NP-completeness of the EDP and VDP problems in undirected planar graphs.

Let us first focus on the case of vertex-disjoint paths. We shall reduce the NP-complete problem planar 3SAT(E3) [13, 14] to the unweighted Min-Sum VDP problem by gap-introducing reduction [20]. Given a planar conjunctive normal formula ϕ, with each clause containing at most three literals and each variable appearing exactly three times, the planar 3SAT(E3) problem asks whether ϕ is satisfiable. Suppose that there are M clauses and N variables in ϕ. Without loss of generality, assume that each variable has two positive occurrences and one negative occurrence (otherwise we can flip all the occurrences of the violated variable to meet the requirement).

Construction of the instance of Min-Sum VDP. We assign the variable gadget G_v in Figure 3 to each variable x_i, and assign the clause gadget G_c in Figure 3 to each clause c_j that possesses three literals. Denote by G_c^2 the gadget resulted by removing vertex l_j^3 from G_c. If c_j has just two literals, then assign c_j the gadget G_c^2. Fix any constant $c > 1$. There is a path from source a_j to sink b_j in the clause gadget, called the *survival path*, which is drawn by thick line in the gadget. The length of the survival path is $\lceil N^c \rceil$.

The dashed line in variable and clause gadgets, which is not part of the gadgets, means that its two endpoints form a terminal pair. The variable gadget G_v has three literal vertices. The left vertex l_i^3 corresponds to the negative occurrence $\neg x_i$, and the right two vertices l_i^1 and l_i^2 correspond to the two positive occurrences x_i. The clause gadget has three literal vertices, corresponding to the three literals in the clause respectively. Note that the literal vertices in G_v and G_c for the same literal are identical. This gives the whole graph $\widetilde{G}$ (not including the survival paths). Graph $\widetilde{G}$ together with all the survival paths form the graph G'. Since ϕ is planar formula, it is clear that G' is a planar graph.

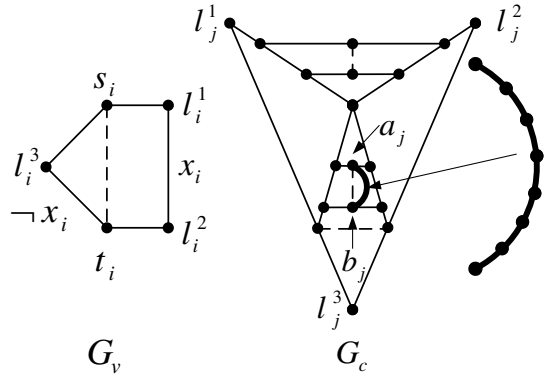

Fig. 3. Gadgets for variables and clauses

Denote by $\widetilde{G}_c$ the gadget resulted by removing the survival path from G_c. We give Lemma 4 and Lemma 5 with respect to G_c. Extending these two lemmas to G_c^2 is trivial.

Lemma 4 ([14])**.** $\widetilde{G}_c$ satisfies the following two properties:

1. $\widetilde{G}_c$ remains feasible for VDP if any at most two literal vertices are removed from $\widetilde{G}_c$, but turns infeasible if all the three literal vertices are removed;
2. If other terminal pair not in $\widetilde{G}_c$ uses one of the paths between any two literal vertices of $\widetilde{G}_c$, $\widetilde{G}_c$ turns infeasible for VDP.

Since there is a survival path in G_c, we get Lemma 5 by generalizing the conclusion in Lemma 4. The existence of the survival path makes G_c having better properties.

Lemma 5. The graph G' resulted from the above reduction is always feasible for VDP. Moreover, ϕ is satisfiable if and only if there exists a feasible solution to G' which does not use any survival path.

Proof. It can be verified that G_c satisfies the following two properties:

1. If any at most two literal vertices are removed, there exists a feasible solution to G_c which does not use any survival path, whereas if all the three literal vertices are removed, then any feasible solution to G_c must use the survival path.
2. If other terminal pair not in G_c uses one of the paths between any two literal vertices of G_c, G_c turns infeasible.

So, no matter whether ϕ is satisfiable, G' is always feasible due to the survival paths.

Suppose that there is a truth assignment satisfying ϕ. Then terminal pair (s_i, t_i) in G_v can be connected by the path corresponding to the literal whose truth value is false. Since ϕ is satisfied, each clause gadget has at least one literal vertex not traversed. By the above property 1, every clause gadget G_c is feasible and no survival path is used.

Conversely, there exists a feasible solution to G' which does not use any survival path. The above property 2 guarantees that the path connecting terminal pair (s_i, t_i) in G_v has to be entirely in G_v. This gives a truth assignment τ for every variable. Suppose that the gadget corresponding to variable x_i is G_v. If s_i and t_i is connected by the path through vertex l_i^3 in G_v, then assign x_i the truth value 1. Otherwise s_i and t_i is connected by the path through vertex l_i^2, and x_i is assigned the truth value 0 in this case. Since the feasible solution to G' does not use any survival path, by the above property 1, there exists at least one literal vertex in G_c which is not used by any variable gadget. Because the literal corresponding to this vertex is true, the clause corresponding to G_c is satisfied. So the formula ϕ is satisfied under the truth assignment τ. □

Theorem 2. Even restricted in feasible instances and in undirected planar graphs the unweighted Min-Sum VDP problem and the unweighted Min-Sum EDP problem cannot be approximated within $\Omega(m^{1-\epsilon})$ for any small constant $\epsilon > 0$, unless P = NP.

Proof. First we prove the theorem for the vertex-disjoint paths case. Let $I_1 = \phi$ be an instance of planar 3SAT(E3). By the construction of the instance of Min-Sum VDP (at the beginning of Section 3), we get an undirected planar graph G'. It is easy to see that the maximum degree of G' is 4. By applying the transformation in Figure 2, we can decrease the degree of non-terminal vertex to 3 and the degree of terminal vertex to 2. Let G be the final graph. Denote by $I_2 = (G, D)$ the resulted instance of Min-Sum VDP.

Since ϕ is a conjunctive normal formula with each clause containing at most 3 literals and each variable appearing exactly 3 times, we know that $N \leq M \leq \frac{3}{2}N$.

If ϕ is satisfiable, by Lemma 5, we have

$$\mathrm{OPT}_{\mathrm{MS}}(I_2) \leq c_1 M + c_2 N \leq c_0 N,$$

where $c_0 = c_1 + c_2$, $c_1 M$ is the total length of vertex-disjoint paths connecting the terminal pairs in all the clause gadgets, and $c_2 N$ is the total length of vertex-disjoint paths connecting the terminal pairs in all the variable gadgets. We charge the length of paths used to connect the introduced terminal pairs in the transformation on the literal vertices to $c_2 N$ (by carefully counting, one can see that $c_1 \leq 55$ and $c_2 \leq 15$).

If ϕ is unsatisfiable, then we have

$$\mathrm{OPT}_{\mathrm{MS}}(I_2) > \lceil N^c \rceil \geq \left(\frac{1}{c_0} N^{c-1}\right) c_0 N.$$

By the construction of the instance I_2 of Min-Sum VDP, we have $m = \Theta(N^{c+1})$, where m is the number of edges in G. So we know $N = \Theta(m^{1/(c+1)})$. This gives the gap $\frac{1}{c_0} N^{c-1} = \Theta\left(m^{1-2/(c+1)}\right)$. By setting $c = \frac{2}{\epsilon} - 1$ for any small constant $\epsilon > 0$, we get the approximation hardness $\Omega(m^{1-\epsilon})$ for unweighted Min-Sum VDP in undirected planar graphs.

Since in graph with maximum degree 2 for terminals and maximum degree 3 for other vertices the vertex-disjoint paths are identical to the edge-disjoint paths, we also obtain approximation hardness $\Omega(m^{1-\epsilon})$ for unweighted Min-Sum EDP in undirected planar graphs. The theorem follows. □

By the well known transformation from undirected graph to directed graph for disjoin paths as shown in Figure 4, we obtain Theorem 3. Moreover, by the proof of Theorem 2, we know that the approximation hardness results in Theorem 2 also apply to the Min-Max Disjoint Paths problem, resulting in Theorem 4. The proof of Theorem 3 and Theorem 4 are omitted.

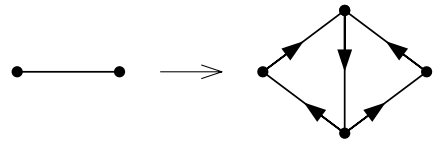

Fig. 4. Transformation from undirected graph to directed graph

Theorem 3. Even restricted in feasible instances and in directed planar graphs, for any small constant $\epsilon > 0$, the unweighted Min-Sum VDP problem and the unweighted Min-Sum EDP problem cannot be approximated within $\Omega(m^{1-\epsilon})$, unless P = NP.

Theorem 4. Even restricted in feasible instances and in (undirected and directed) planar graphs, for any small constant $\epsilon > 0$, the unweighted Min-Max VDP problem and the unweighted Min-Max EDP problem cannot be approximated within $\Omega(m^{1-\epsilon})$, unless P = NP.

Since assuming $m = \Omega(n)$ does not lose generality, the unweighted Min-Sum Disjoint Paths problem and the unweighted Min-Max Disjoint Paths problem also have approximation hardness $\Omega(n^{1-\epsilon})$ for any small constant $\epsilon > 0$ even in planar graphs, where n is the number of vertices in graph G.

4 THE RANDOMIZED APPROXIMATION ALGORITHMS

The complexity and approximation hardness of the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem show that efficient heuristics for the problems may be useful in practice. Several heuristics and experimental results for Min-Sum EDP are given in [1]. Other than experimental results, we give for the first time an efficient approximation algorithm with guaranteed performance ratio for the problems, allowing that some constraints are slightly violated. Although the problems are very hard in terms of complexity and approximation hardness, we can get an $(O(\log k/\log\log k), O(\log n/\log\log n))$-approximation for the unweighted Min-Max EDP problem and an $(O(1), O(\log n/\log\log n))$-approximation for the weighted Min-Sum EDP problem. Our algorithm is based on randomized rounding.

Some Concepts and the Notation. The approximation algorithm with a bi-factor performance ratio is often called a *bicriteria approximation algorithm*. In the bi-factor of our algorithm, the first parameter is the approximation ratio, and the second parameter is the maximum number of paths per edge in the solution, called congestion. Recall that the term congestion has been defined in Section 1.3.

Let p be any simple path. Define $w(p) = \sum_{e\in p} w_e$ as the *length* of path p. In the unweighted case, $w(p)$ is just $|p|$, the number of edges in path p. Let S be a solution (We view S as a set of paths) to some disjoint paths problem. Define $\text{LEN}_{\max}(S) = \max_{p\in S}\{w(p)\}$ as the *length of the longest path* in solution S'. Defined $\text{LEN}_{\text{tot}}(S) = \sum_{p\in S} w(p)$ as the *total length* [2] of solution S.

For an instance I of the unweighted Min-Max EDP problem and its solution S, define the *stretch* of solution S as $\frac{\text{LEN}_{\max}(S)}{\text{OPT}_{\text{MM}}(I)}$. Note that the stretch of solution S is just *its* approximation ratio. For the sake of analysis of our algorithm, we need the notion of the *stretch of S against L* for some quantity L, which is defined as $\frac{\text{LEN}_{\max}(S)}{L}$.

For an instance I of the weighted Min-Sum EDP problem and its solution S, define the *stretch* of solution S as $\frac{\text{LEN}_{\text{tot}}(S)}{\text{OPT}_{\text{MS}}(I)}$. Also, for the sake of analysis of our algorithm, we need the notion of the *stretch of S against* OPT_f for some quantity OPT_f, which is defined as $\frac{\text{LEN}_{\text{tot}}(S)}{\text{OPT}_f}$.

[2] We define the total length of solution S as $\sum_{p\in S} w(p)$ in terms of *paths*. Another definition of the total length is $\sum_{e:\text{ contained in }S} w_e$, which is defined in terms of *edges*. For a (bicriteria) solution S with congestion larger than one, $\sum_{e:\text{ contained in }S} w_e$ is obviously at most $\sum_{p\in S} w(p)$, implying that Theorem 6 still holds if we adopt the seemingly more traditional definition $\sum_{e:\text{ contained in }S} w_e$ of total length.

Note that for the unweighted Min-Max EDP problem and the weighted Min-Sum EDP problem, the congestion of the solutions found by our algorithm is only $O(\log n/\log\log n)$, a super-constant, while the stretch of our solutions is a constant or a super-constant.

4.1 LP Relaxation and Approximation Algorithm for Unweighted Min-Max EDP

First we consider the unweighted Min-Max EDP problem. The graph in the instance can be directed or undirected. The fractional linear program for unweighted Min-Max EDP is shown as $(LP1)$.

$$
\begin{aligned}
(LP1) \qquad & \min\ L && && (1)\\
\text{s. t.}\quad & \\
& \sum_{p:\ e\in p} f(p) \ \le\ 1, && \forall e\in E && (2)\\
& \sum_{p\in P_i} f(p) \ =\ 1, && \forall i\in [k] && (3)\\
& \sum_{p\in P_i} |p| f(p) \ \le\ L, && \forall i\in [k] && (4)\\
& f(p) \ \ge\ 0, && \forall p\in \mathcal{P} &&\\
& L \ \ge\ 0 &&
\end{aligned}
$$

The linear program $(LP1)$ view the disjoint paths problem as a multicommodity flow problem. In the linear program $(LP1)$, P_i denotes the set of all the possible simple paths from s_i to t_i, and $\mathcal{P}$ is the union of all P_i for $1 \le i \le k$. For each path $p \in P_i$, we define a variable $f(p) \in [0,1]$ which denotes the flow value on p for terminal pair (s_i, t_i). Consider the integral version (that is, $f(p) \in \{0,1\}$ and $L \in \{1,2,\ldots,m\}$) of $(LP1)$. Then (2) specifies the capacity constraint for every edge, (3) specifies the connectivity requirement for every terminal pair, and (4) specifies that the maximum length of flow path used for every terminal pair should not exceed L. The objective function (1) is to minimize the length of the longest path that has ever been used. The symbol $[k]$ in constraint (3) and (4) stands for the set $\{1,2,\ldots,k\}$.

Notice that although $(LP1)$ has exponential size, we can obtain a solution to $(LP1)$ in polynomial time, since there is a polynomial-size linear program equivalent to $(LP1)$. The polynomial-size linear program for unweighted Min-Max EDP can be obtained by the method introduced in [5]. We can first solve the equivalent polynomial-size linear program, then get an optimal solution to $(LP1)$ by using the flow decomposition method.

For completeness, we give the polynomial-size integer linear program formulation for the unweighted Min-Max EDP problem, as shown in (IP). Suppose that the underlying graph G is directed.

$$
\begin{aligned}
(IP) \qquad & \min L && && (5)\\
\text{s. t.}\quad & \\
& \sum_{i\in[k]} f^i_{u,v} \;\le\; 1, && \forall (u,v)\in E && (6)\\
\sum_{(u,v)\in E} f^i_{u,v} - & \sum_{(v,w)\in E} f^i_{v,w} \;\ge\; 0, && \forall i\in[k], \forall v\in V && (7)\\
& \sum_{(s_i,v)\in E} f^i_{s_i,v} \;=\; 1, && \forall i\in[k] && (8)\\
& \sum_{(u,v)\in E} f^i_{u,v} \;\le\; L, && \forall i\in[k] && (9)\\
& f^i_{u,v} \;\in\; \{0,1\}, && \forall i\in[k], \forall (u,v)\in E\\
& L \;\in\; \{0,1,2,\ldots,m\}
\end{aligned}
$$

For each edge $(u,v)\in E$ and for each terminal pair (s_i,t_i), we define a variable $f^i_{u,v}\in\{0,1\}$ to denote the capacity used on edge (u,v) by terminal pair (s_i,t_i); and for each terminal pair (s_i,t_i), we add an edge (t_i,s_i) to E. Only variable $f^i_{t_i,s_i}$ is defined for edge (t_i,s_i). Let $F=\{(t_1,s_1),\ldots,(t_k,s_k)\}$. Every edge $e\in E$ has unit capacity. Then (6) denotes the capacity constraint for every edge, (7) states that the total flow out of vertex v is at most the total flow into it, and (8) states that for each terminal pair there must be a flow with value 1. Note that if (7) holds for every vertex, it actually holds with equality, implying the flow conservation at each vertex. By the integrality of the linear program, (9) states that the length of path for every terminal pair (s_i,t_i) should not exceed L. Then the objective function (5) is to minimize the maximum length of paths ever used. Furthermore, since every edge has unit capacity, the paths in the solution must be edge-disjoint.

The linear program formulation for the undirected graph is similar to the directed case. For each edge $(u,v)\in E\backslash F$, we define two variables $f^i_{u,v}$ and $f^i_{v,u}$ to denote the flow value from u to v and that from v to u on edge (u,v) for terminal pair (s_i,t_i), respectively. For the added edge $(t_i,s_i)\in F$, we only define one variable $f^i_{t_i,s_i}$.

Suppose we have an optimal solution (f,L) to the fractional relaxation of (IP). Then we can get an (optimal) solution to $(LP1)$ by using the flow decomposition method. For simplicity, we assume that the underlying graph G is directed. Fix any terminal pair (s_i,t_i). Let G^i be the graph $(V(G),E(G))$ with capacities $\{f^i_{u,v}\}$ defined on edges. By the reachability method, we can extract a simple path p from s_i to t_i along nonzero-capacity edges. The flow value $f(p)$ is defined as the minimum capacity of edges in path p. For each edge $(u,v)\in p$, decrease its capacity to $f^i_{u,v}-f(p)$; then delete all the edges with zero capacity from G^i. That is, we use

up at least one edge once we find an (s_i, t_i)-path. The procedure is repeated until t_i is unreachable from s_i in the current G^i. Note that the number of (s_i, t_i)-paths with positive flow value obtained in this manner is polynomial in n, the number of vertices in G.

Now we give the approximation algorithm for the unweighted Min-Max EDP problem, as shown in Algorithm $\mathcal{A}$. Algorithm $\mathcal{A}$ is a simple randomized approximation algorithm. It first solves $(LP1)$ to get an optimal solution (f, L), then rounds f to an integral solution by randomly picking a path $p \in P_i$ as casting a $|P_i|$-side die, whose each side denotes a path in P_i. To improve the probability of obtaining a solution with guaranteed performance, Algorithm $\mathcal{A}$ repeats the above randomized rounding procedure for sufficient times.

Algorithm $\mathcal{A}$ for unweighted Min-Max EDP

1. **if** $(LP1)$ has no solution, **then return** "infeasible" and **halt**.
2. Compute an optimal solution (f, L) to $(LP1)$.
3. **repeat** the following steps from 4 to 8 for $r = \lceil 2 \log n \rceil$ times. In the j^{th} iteration, **do**
4. **let** $S_j \leftarrow \emptyset$.
5. **for each** terminal pair (s_i, t_i) **do**
6. Choose $p \in P_i$ exclusively at random with probability $f(p)$.
7. **let** $S_j \leftarrow S_j \cup \{p\}$.
8. **end**
9. **end**
10. Find a solution from $\{S_j\}$ such that its stretch against L is at most α and its congestion is at most β. If there is no such solution, then pick any one from $\{S_j\}$. Let S be the selected solution.
11. **return** S.

The parameters α and β in Algorithm $\mathcal{A}$ will be given in the proof of Lemma 6.

Lemma 6. Let I be an instance of the unweighted Min-Max EDP problem, and S' be the solution found in any iteration of Algorithm $\mathcal{A}$ on instance I. Then S' connects all the terminal pairs. Moreover, S' satisfies the following properties with probability $> \frac{1}{2}$:

1. The stretch of S' is at most $O(\log k / \log \log k)$ (i.e., the length of the longest path in S' is at most $O(\log k / \log \log k)\mathrm{OPT}(I)$), and
2. The congestion of S' is at most $O(\log n / \log \log n)$ (i.e., the maximum number of paths per edge in S' is at most $O(\log n / \log \log n)$).

Proof. Consider the iteration that generates S'. Since we pick a path $p \in P_i$ as casting a $|P_i|$-side die for each terminal pair (s_i, t_i), obviously S' connects all the terminal pairs. Then we turn to the property 1 in the lemma.

Fix any terminal pair (s_i, t_i). For every edge e contained in some path in P_i, define random variable Z_i^e to denote whether edge e is used to connect terminal pair (s_i, t_i). If e is in some path $p \in P_i$ which is selected by Algorithm $\mathcal{A}$, then $Z_i^e = 1$, otherwise $Z_i^e = 0$. Then define random variable

$$Z_i = \sum_{\substack{e:\, e \in q \\ q \in P_i}} Z_i^e$$

to be the length of the path in S' connecting (s_i, t_i).

By definition of Z_i^e, we have

$$\mathrm{E}[Z_i^e] = \sum_{\substack{p:\, e \in p \\ p \in P_i}} f(p).$$

So,

$$\mathrm{E}[Z_i] = \sum_{\substack{e:\, e \in q \\ q \in P_i}} \mathrm{E}[Z_i^e] = \sum_{\substack{e:\, e \in q \\ q \in P_i}} \sum_{\substack{p:\, e \in p \\ p \in P_i}} f(p) = \sum_{p \in P_i} f(p)|p| \le L,$$

where the last inequality is due to constraint (4).

About the fractional optimum L of $(LP1)$, we have

$$L \ge \sum_{p \in P_i} f(p)|p| \ge \min_{p \in P_i}\{|p|\} \sum_{p \in P_i} f(p) = \min_{p \in P_i}\{|p|\} \ge 1$$

by constraints (4) and (3).

Define $\alpha = \frac{2 \ln k}{\ln \ln k}$. Set $\delta = \alpha \cdot \frac{L}{\mathrm{E}[Z_i]} - 1$. Since $\mathrm{E}[Z_i] \le L$, we have $\delta > 0$. By the Chernoff bound, we obtain

$$\begin{aligned}
\Pr[Z_i > \alpha L] &= \Pr[Z_i > (1+\delta)\mathrm{E}[Z_i]] \\
&< \left(\frac{e}{1+\delta}\right)^{(1+\delta)\mathrm{E}[Z_i]} \\
&= \left(\frac{e \cdot \mathrm{E}[Z_i] \ln \ln k}{2 \ln k \cdot L}\right)^{2 \ln k \cdot L / \ln \ln k} \\
&\le \left(\frac{1}{4k}\right)^{L} \\
&\le \frac{1}{4k}
\end{aligned}$$

when k is sufficiently large, where the last inequality holds since $L \ge 1$.

Define random variable $Z = \mathrm{LEN}_{\max}(S')$, and denote by B_1 the event that $Z > \alpha \cdot \mathrm{OPT}(I)$. Then we know that

$$\begin{aligned}\Pr[B_1] &\leq \Pr[Z > \alpha L] \\ &= \Pr[\exists i, Z_i > \alpha L] \\ &\leq \sum_{i=1}^{k} \Pr[Z_i > \alpha L] \\ &\leq \frac{1}{4}. \end{aligned} \tag{10}$$

There may be some edges used for many times in solution S'. For the sake of the analysis to property 2, fix any edge $e \in E$. For each path $p \in \mathcal{P}$, define random variable X_p as 1 if p is chosen and as 0 otherwise. Then define random variable

$$X_e = \sum_{p:\, e \in p} X_p$$

as the number of times that e is used in the procedure of randomized rounding for all terminal pairs. Then we know that $\mathrm{E}[X_p] = f(p)$ and

$$\mathrm{E}[X_e] = \sum_{p:\, e \in p} \mathrm{E}[X_p] = \sum_{p:\, e \in p} f(p) \leq 1$$

by constraint (2).

Define $\beta = \frac{3 \ln n}{\ln \ln n}$. Set $\delta = \frac{\beta}{E[X_e]} - 1$. Since $\mathrm{E}[X_e] \leq 1$, we have that $\delta > 0$. Again by the Chernoff bound, we have

$$\begin{aligned}\Pr[X_e > \beta] &= \Pr[X_e > (1+\delta)\mathrm{E}[X_e]] \\ &< \left(\frac{e}{1+\delta}\right)^{(1+\delta)\mathrm{E}[X_e]} \\ &= \left(\frac{e \cdot \mathrm{E}[X_e] \ln \ln n}{3 \ln n}\right)^{3 \ln n / \ln \ln n} \\ &\leq \left(\frac{e \cdot \ln \ln n}{3 \ln n}\right)^{3 \ln n / \ln \ln n} \\ &< \frac{1}{4n^2} \end{aligned}$$

when n is sufficiently large.

Denote by B_2 the event that there is an edge $e \in E$ such that $X_e > \beta$. Then we know

$$\Pr[B_2] \leq \sum_{e \in E} \Pr[X_e > \beta] < \frac{1}{4}. \tag{11}$$

By inequalities (10) and (11), we have $\Pr[B_1 + B_2] < 1/2$. This gives the lemma. □

Theorem 5. As long as $(LP1)$ has a fractional feasible solution, in polynomial time Algorithm $\mathcal{A}$ outputs a solution S that connects all the terminal pairs. Moreover, S satisfies the following properties

1. the stretch of S is at most $O(\log k / \log\log k)$,
2. the congestion of S is at most $O(\log n / \log\log n)$

with high probability.

Proof. Since $(LP1)$ is feasible, Algorithm $\mathcal{A}$ will output a (bicriteria) solution S to the problem. By repeating the procedure of randomized rounding for $\lceil 2\log n \rceil$ times, we get a solution to the problem satisfying all the properties in the theorem with probability at least $1 - 1/n^2$.

An optimal solution to $(LP1)$ can be computed in polynomial time. Note that when we obtain an optimal solution to $(LP1)$ by the flow decomposition method, the number of (s_i, t_i)-paths with positive flow value is polynomial in n for each terminal pair (s_i, t_i). This shows that the randomized rounding procedure of Algorithm $\mathcal{A}$ takes polynomial time. Therefore the running time of Algorithm $\mathcal{A}$ is polynomial. This completes the proof. □

Remarks about Algorithm $\mathcal{A}$. If the instance of Min-Max EDP is feasible for EDP, then $(LP1)$ must have a fractional feasible solution. In this case Algorithm $\mathcal{A}$ outputs a solution to the problem with the guaranteed stretch and congestion. The subtle case is that, even if the instance of Min-Max EDP is *not* feasible for EDP, Algorithm $\mathcal{A}$ still may output a solution with the guaranteed stretch and congestion, provided that $(LP1)$ has a fractional feasible solution. Since solving $(LP1)$ (e.g., by the ellipsoid algorithm) tells us whether it has a fractional feasible solution, Algorithm $\mathcal{A}$ gets around the issue of deciding the NP-complete EDP problem.

Corollary 1. For feasible instances of the unweighted Min-Max EDP problem, Algorithm $\mathcal{A}$ outputs in polynomial time a bicriteria solution with stretch $O\left(\frac{\log k}{\log\log k}\right)$ and congestion $O\left(\frac{\log n}{\log\log n}\right)$. Moreover, if Algorithm $\mathcal{A}$ returns infeasible, then the instance of the unweighted Min-Max EDP problem is not feasible for EDP.

4.2 LP Relaxation and Approximation Algorithm for Weighted Min-Sum EDP

The fractional linear program for the weighted Min-Sum EDP problem is similar to that of the unweighted Min-Max EDP problem, as shown in $(LP2)$.

$$
\begin{aligned}
(LP2) \qquad \min \ & \sum_{p\in\mathcal{P}} w(p)f(p) \\
\text{s. t.} \quad & \\
& \sum_{p:\, e\in p} f(p) \ \leq \ 1, && \forall e \in E \\
& \sum_{p\in P_i} f(p) \ = \ 1, && \forall i \in [k] \\
& f(p) \ \geq \ 0, && \forall p \in \mathcal{P}
\end{aligned}
$$

Our approximation algorithm for the weighted Min-Sum EDP problem is very similar to Algorithm $\mathcal{A}$ for the unweighted Min-Max EDP problem, except that in step 10 the algorithm should find a solution from $\{S_j\}$ such that its stretch is at most α against $\mathrm{OPT}_f(LP2)$ (instead of L) and its congestion is at most β. Of course, step 1 and step 2 in Algorithm $\mathcal{A}$ should be modified in the obvious way. For clarity, let us call the algorithm for weighted Min-Sum EDP Algorithm $\mathcal{B}$. The parameters α and β in Algorithm $\mathcal{B}$ will be given in Lemma 7.

Lemma 7. Let I be an instance of the weighted Min-Sum EDP problem, and S' be the solution found in any iteration of Algorithm $\mathcal{B}$ on instance I. Then S' connects all the terminal pairs. Moreover, S' satisfies the following properties with probability $> \frac{1}{2}$:

1. The stretch of S' is at most $O(1)$ (i.e., the total length $\mathrm{LEN}_{\mathrm{tot}}(S')$ is at most $O(1)\mathrm{OPT}_{\mathrm{MS}}(I)$), and
2. The congestion of S' is at most $O(\log n/\log\log n)$ (i.e., the maximum number of paths per edge in S' is at most $O(\log n/\log\log n)$).

Proof. We only prove property 1 in the lemma. Define random variable Y to be the total length of S'. Then,

$$\mathrm{E}[Y] = \sum_{p\in\mathcal{P}} w(p)\Pr[p\in S] = \sum_{p\in\mathcal{P}} w(p)f(p) = \mathrm{OPT}_f(I) \leq \mathrm{OPT}_{\mathrm{MS}}(I).$$

Notice that the expected total length of S' is always lower than $\mathrm{OPT}_{\mathrm{MS}}(I)$. By Markov's inequality, for any constant $c > 0$, the probability of $Y > c\cdot\mathrm{OPT}_{\mathrm{MS}}(I)$ is upper bounded by $1/c$. We choose $\alpha = 4$ so that the probability of $Y > 4\cdot\mathrm{OPT}_{\mathrm{MS}}(I)$ is at most $\frac{1}{4}$. Combining with the analysis for edge congestion in Lemma 6 completes the proof of the lemma, where the parameter β is set to be $\frac{3\ln n}{\ln\ln n}$. □

The proof of the following Theorem 6 is straightforward and is omitted.

Theorem 6. As long as $(LP2)$ has a fractional feasible solution, in polynomial time Algorithm $\mathcal{B}$ outputs a solution S that connects all the terminal pairs. Moreover, S satisfies the following properties

1. the stretch of S is at most $O(1)$,
2. the congestion of S is at most $O(\log n / \log \log n)$

with high probability.

5 CONCLUSIONS

We systematically study the complexity and approximation hardness of the Min-Sum Disjoint Paths problem and the Min-Max Disjoint Paths problem. We give for the first time a randomized bicriteria approximation algorithm for the weighted Min-Sum EDP problem and the unweighted Min-Max EDP problem in general graphs. We suspect that the weighted Min-Sum Disjoint Paths problem is $\mathrm{FP}^{\mathrm{NP}}$-complete even in more constrained graphs. In fact, Min-Sum Disjoint Paths can be proved to be still $\mathrm{FP}^{\mathrm{NP}}$-complete in 3-regular graphs by slight modifications of the gadgets used in Theorem 1. In addition, can one get an approximation algorithm for Min-Sum Disjoint Paths or Min-Max Disjoint Paths, with slightly relaxed approximation ratio but constant bound on the congestion? Both the problems are interesting.

Acknowledgement

The authors are very grateful to the anonymous reviewer for the valuable comments, which help make the results of the paper more clear and improve the presentation of the paper. The authors also thank the anonymous reviewer for showing us some related references.

REFERENCES

[1] Brandes, U.—Neyer, G.—Wagner, D.: Edge-Disjoint Paths in Planar Graphs with Short Total Length. Technical Report, in Konstanzer Schriften in Mathematik und Informatik, No. 19, 1996.

[2] Chekuri, C.—Khanna, S.: Edge Disjoint Paths Revisited. In: Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms (SODA), Baltimore, MD, USA, 2003, pp. 628–637.

[3] Colin de Verdière, E.—Schrijver, A.: Shortest Vertex-Disjoint Two-Face Paths in Planar Graphs. In: Proceedings of the 25th Annual Symposium on Theoretical Aspects of Computer Science (STACS), Bordeaux, France 2008, pp. 181–192.

[4] Fortune, S.—Hopcroft, J. E.—Wyllie, J.: The Directed Subgraph Homeomorphism Problem. Theoretical Computer Science, Vol. 10, 1980, pp. 111–121.

[5] Garg, N.—Vazirani, V.—Yannakakis, M.: Approximate Max-Flow Min-(Multi)cut Theorems and Their Applications. SIAM Journal on Computing, Vol. 25, 1996, No. 2, pp. 235–251.

[6] Itai, A.—Perl, Y.—Shiloach, Y.: The Complexity of Finding Maximum Disjoint Paths with Length Constraints. Networks, Vol. 12, 1982, No. 3, pp. 277–286.

[7] Karp, R.: On the Computational Complexity of Combinatorial Problems. Networks, Vol. 5, 1975, pp. 45–8.

[8] Kleinberg, J.: Approximation Algorithms for Disjoint Paths Problems. Ph. D. Thesis, Department of EECS, MIT, Cambridge, MA, 1996.

[9] Kleinberg, J.: An Approximation Algorithm for the Disjoint Paths Problem in Even-Degree Planar Graphs. In: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science, Pittsburgh, PA, USA 2005, pp. 627–636.

[10] Kobayashi, Y.—Sommer, C.: On Shortest Disjoint Paths in Planar Graphs. Discrete Optimization, Vol. 7, 2010, No. 4, pp. 234–245.

[11] Korte, B.—Lovász, L.–Prömel, H.—Schrijver, A.: Paths, Flows, and VLSI Layout. Springer-Verlag, Berlin Heidelberg 1990.

[12] Li, C.—McCormick, T.—Simchi-Levi, D.: The Complexity of Finding two Disjoint Paths with Min-max Objective Function. Discrete Applied Mathematics, Vol. 26, 1990, pp. 105–115.

[13] Lichtenstein, D.: Planar Formulae and Their Uses. SIAM Journal on Computing, Vol. 11, 1982, pp. 329–343.

[14] Middendorf, M.—Pfeiffer, F.: On the Complexity of the Disjoint Paths Problem. Combinatorica, Vol. 13, 1993, No. 1, pp. 97–107.

[15] Papadimitriou, C.: Computational Complexity. Addison-Wesley Publishing Company Inc. 1994.

[16] Raghavan, P.—Thompson, C.: Randomized Rounding: A Technique for Provably Good Algorithms and Algorithmic Proofs. Combinatorica, Vol. 7, 1987, No. 4, pp. 365–374.

[17] Scheffler, P.: A Practical Linear Time Algorithm for Disjoint Paths in Graphs with Bounded Tree-Width. Technical Report 396, Technische Universität Berlin 1994.

[18] Suurballe, J.—Tarjan, R.: A Quick Method for Finding Shortest Pairs of Disjoint Paths. Networks, Vol. 14, 1984, pp. 325–336.

[19] Tragoudas, S.—Varol, Y.: Computing Disjoint Path with Length Constraints. In: Proccedings of the 22nd International Workshop of Graph-Theoretic Concepts in Computer Science (WG), Cadenabbia (Como), Italy 1996, pp. 375–389.

[20] Vazirani, V.: Approximation Algorithms. 2nd Edition, Springer-Verlag Berlin Heidelberg 2003.

[21] Yang, B.—Zheng, S. Q.: Finding Min-Sum Disjoint Shortest Paths From a Single Source to all Pairs of Destinations. In: Proceedings of the 3rd International Conference on Theory and Applications of Models of Computation (TAMC), Beijing, China 2006, pp. 206–216.

[22] Zhang, P.—Zhao, W.: On the Complexity and Approximation of the Min-Sum and Min-Max Disjoint Paths Problems. In: Proceedings of the 1st International Symposium on Combinatorics, Algorithms, Probabilistic and Experimental Methodologies (ESCAPE), Hangzhou, China 2007, pp. 70–81.

Peng Zhang received his B. Sc. degree in 1999 and M. Sc. degree in 2004 in computer science from School of Computer Science and Technology, Shandong University, China, and his Ph. D. degree in computer science in 2007 from Institute of Software, Chinese Academy of Sciences. Now he is an Associate Professor of computer science at School of Computer Science and Technology, Shandong University, China. His research interests include approximation algorithms, combinatorial optimization, and computational complexity.

Wenbo Zhao received his B. Sc. degree in mathematics in 2004 from Nanjing University, China, and his M. Sc. degree in mathematics in 2007 from Institute of Software, Chinese Academy of Sciences. Currently he is a PhD student at University of California, San Diego. His research interests include combinatorial optimization, approximation algorithms, and bioinformatics.

Daming Zhu received his Ph. D. degree in computer science in 1999 from Institute of Computing Technology, Chinese Academy of Sciences. He is currently a professor and a Ph. D. supervisor of computer science at School of Computer Science and Technology, Shandong University, China, and a senior member of China Computer Federation. His main research interests include the design and analysis of algorithms, computational complexity, and bioinformatics.