Finding Two Edge-Disjoint Paths with Length Constraints

Leizhen Cai and Junjie Ye(✉)

Department of Computer Science and Engineering,
The Chinese University of Hong Kong, Shatin, Hong Kong SAR, China
{lcai,jjye}@cse.cuhk.edu.hk

Abstract. We consider the problem of finding, for two pairs (s_1, t_1) and (s_2, t_2) of vertices in an undirected graph, an (s_1, t_1)-path P_1 and an (s_2, t_2)-path P_2 such that P_1 and P_2 share no edges and the length of each P_i satisfies L_i, where $L_i \in \{\leq k_i, = k_i, \geq k_i, \leq \infty\}$. We regard k_1 and k_2 as parameters and investigate the parameterized complexity of the above problem when at least one of P_1 and P_2 has a length constraint (note that $L_i =$ "$\leq \infty$" indicates that P_i has no length constraint). For the nine different cases of (L_1, L_2), we obtain FPT algorithms for seven of them. Our algorithms uses random partition backed by some structural results. On the other hand, we prove that the problem admits no polynomial kernel for all nine cases unless $NP \subseteq coNP/poly$.

Keywords: Edge-disjoint paths · Random partition · Parameterized complexity · Kernelization

1 Introduction

Disjoint paths in graphs are fundamental and have been studied extensively in the literature. Given k pairs of *terminal vertices* (s_i, t_i) for $1 \leq i \leq k$ in an undirected graph G, the classical Edge-Disjoint Paths problem asks whether G contains k pairwise edge-disjoint paths P_i between s_i and t_i for all $1 \leq i \leq k$. The problem is NP-complete as shown by Even et al. [8], but is solvable in time $O(mn)$ by network flow [16] if all vertices s_i (resp., t_i) are the same vertex s (resp., t). When we regard k as a parameter, a celebrated result of Robertson and Seymour [17] on vertex-disjoint paths can be used to obtain an FPT algorithm for Edge-Disjoint Paths. On the other hand, Bodlaender et al. [4] have shown that Edge-Disjoint Paths admits no polynomial kernel unless $NP \subseteq coNP/poly$.

In this paper, we study Edge-Disjoint Paths with length constraints L_i on (s_i, t_i)-paths P_i and focus on the problem for two pairs of terminal vertices. The length constraints $L_i \in \{\leq k_i, = k_i, \geq k_i, \leq \infty\}$ indicate that the length of P_i need to satisfy L_i. We regard k_1 and k_2 as parameters, and study the parameterized complexity of the following problem.

Partially supported by GRF grant CUHK410212 of the Research Grants Council of Hong Kong.

© Springer-Verlag GmbH Germany 2016
P. Heggernes (Ed.): WG 2016, LNCS 9941, pp. 62–73, 2016.
DOI: 10.1007/978-3-662-53536-3_6

EDGE-DISJOINT (L_1, L_2)-PATHS
INSTANCE: Graph $G = (V, E)$, two pairs (s_1, t_1) and (s_2, t_2) of vertices.
QUESTION: Does G contain (s_i, t_i)-paths P_i for $i = 1, 2$ such that P_1 and P_2 share no edge and the length of P_i satisfies L_i?

There are nine different length constraints on two paths (note that EDGE-DISJOINT $(\leq \infty, \leq \infty)$-PATHS puts no length constraint on two paths). For instance, EDGE-DISJOINT $(= k_1, \leq \infty)$-PATHS requires that $|P_1| = k_1$ but P_2 has no length constraint, and EDGE-DISJOINT $(= k_1, \geq k_2)$-PATHS requires that $|P_1| = k_1$ and $|P_2| \geq k_2$.

Related Work. EDGE-DISJOINT (L_1, L_2)-PATHS has been studied under the framework of classical complexity. Ohtsuki [15], Seymour [18], Shiloah [20], and Thomasssen [21] independently gave polynomial-time algorithms for EDGE-DISJOINT $(\leq \infty, \leq \infty)$-PATHS. Tragoudas and Varol [22] proved the NP-completeness of EDGE-DISJOINT $(\leq k_1, \leq k_2)$-PATHS, and Eilam-Tzoreff [7] showed the NP-completeness of EDGE-DISJOINT $(\leq k_1, \leq \infty)$-PATHS even when k_1 equals the (s_1, t_1)-distance. For EDGE-DISJOINT (L_1, L_2)-PATHS with $L_1 = k_1$ or $\geq k_1$ (same for $L_2 = k_2$ or $\geq k_2$), we can easily establish its NP-completeness by reductions from the classical HAMILTONIAN PATH problem.

As for the parameterized complexity, there are a few results in connection with our EDGE-DISJOINT (L_1, L_2)-PATHS. Golovach and Thilikos [13] obtained an $2^{O(kl)} m \log n$-time algorithm for EDGE-DISJOINT PATHS when every path has length at most l. For a single pair (s, t) of vertices, an (s, t)-path of length exactly l can be found in time $O(2.6181^l m \log^2 n)$ [9,19] and $O^*(2.5961^l)$ [23]. (Note that l-PATH that finds a path of length l can be solved in time $O(2.6181^l n \log^2 n)$ [9,19].) For the problem of finding an (s, t)-path of length at least l, Bodlaender [1] derived an $O(2^{2l}(2l)!n + m)$-time algorithm, Gabow and Nie [12] designed an $l^l 2^{O(l)} mn \log n$-time algorithm, and FPT algorithms of Fomin et al. [10] for cycles and paths can be adapted to yield an $8^{l+o(l)} m \log^2 n$-time algorithm.

Our Contributions. In this paper, we investigate the parameterized complexity of EDGE-DISJOINT (L_1, L_2)-PATHS for the nine different length constraints and have obtained FPT algorithms for seven of them (see Table 1 for a summary).

In particular, we use random partition in an interesting way to obtain FPT algorithms for EDGE-DISJOINT $(= k_1, \leq \infty)$-PATHS and EDGE-DISJOINT $(= k_1, \geq k_2)$-PATHS. This is achieved by bounding the number of some special edges, called "nearby-edges", in the two paths P_1 and P_2 by a function of k_1 and k_2 alone. We also consider polynomial kernels and prove that all nines cases admit no polynomial kernel unless $NP \subseteq coNP/poly$.

Notation and Definitions. All graphs in the paper are simple undirected connected graphs. For a graph G, we use $V(G)$ and $E(G)$ to denote its vertex set and edge set respectively, and n and m, respectively, are numbers of vertices and edges of G. For two vertices s and t, the distance between s and t is denoted by $d(s, t)$.

Table 1. Running times of FPT algorithms for Edge-Disjoint (L_1, L_2)-Paths with length constraints $L_i \in \{\leq k_i, = k_i, \geq k_i, \leq \infty\}$ for $i = 1, 2$. Note that $r_1 = k_1 + k_2, r_2 = k_1^2 + 5k_2$, and $r_3 = k_2^2 + 5k_1$.

Constraints	$\|P_2\| \leq k_2$	$\|P_2\| = k_2$	$\|P_2\| \geq k_2$	$\leq \infty$
$\|P_1\| \leq k_1$	$O(2.01^{r_1} m \log n)$		$O(2.01^{r_2} m \log^3 n)$	$O(2.01^{k_1^2} m \log n)$
$\|P_1\| = k_1$	$O(5.24^{r_1} m \log^3 n)$			$O(2.01^{k_1^2} m \log^3 n)$
$\|P_1\| \geq k_1$	$O(2.01^{r_3} m \log^3 n)$		Open	

An instance (I, k) of a parameterized problem Π consists of two parts: an input I and a parameter k. We say that a parameterized problem Π is fixed-parameter tractable (FPT) if there is an algorithm solving every instance (I, k) in time $f(k)|I|^{O(1)}$ for some computable function f. A kernelization algorithm for a parameterized problem Π maps an instance (I, k) in time polynomial in $|I| + k$ into a smaller instance (I', k') such that (I, k) is a yes-instance iff (I', k') is a yes-instance and $|I'| + k' \leq g(k)$ for some computable function g. Problem Π has a polynomial kernel if $g(k)$ is a polynomial function.

For simplicity, we write $O(2.01^{f(k)})$ for $2^{f(k)+o(f(k))}$ as the latter is $O((2 + \epsilon)^{f(k)})$ for any constant $\epsilon > 0$ and we choose $\epsilon = 0.01$. In particular, $2^k k^{O(\log k)} = 2^{k+O(\log^2 k)} = O(2.01^k)$.

In the rest of the paper, we present FPT algorithms for seven cases in Sect. 2, and show the nonexistence of polynomial kernels in Sect. 3. We conclude with some open problems in Sect. 4.

2 FPT Algorithms

Random partition provides a natural tool for finding edge-disjoint (L_1, L_2)-paths in a graph G: We randomly partition edges of G to form two graphs G_1 and G_2, and then independently find paths P_1 in G_1 (resp., P_2 in G_2) whose lengths satisfy L_1 (resp., L_2).

When our problem satisfies the following two conditions, the above approach yields a randomized FPT algorithm and can typically be derandomized by universal sets.

1. Whenever G has a solution, the probability of "G_1 contains required P_1 and G_2 contains required P_2" is bounded below by a function of k_1 and k_2 alone.
2. It takes FPT time to find required paths P_1 in G_1 and P_2 in G_2.

Indeed, straightforward applications of the above method yield FPT algorithms for Edge-Disjoint (L_1, L_2)-Paths when $L_i \in \{\leq k_i, = k_i\}$ for $i = 1, 2$.

Theorem 1. Edge-Disjoint (L_1, L_2)-Paths *can be solved in* $O(2.01^{k_1+k_2} m \log n)$ *time for* $(L_1, L_2) = (\leq k_1, \leq k_2)$*, and* $O(5.24^{k_1+k_2} m \log^3 n)$ *time for* $(L_1, L_2) = (\leq k_1, = k_2)$ *or* $(= k_1, = k_2)$.

Proof. Let $r = k_1 + k_2$. We randomly color each edge by color 1 or 2 with probability 1/2 to define a random partition of edges. Denote by $G_i, i = 1, 2$, the graph consisting of edges of color i. Then for all three cases of (L_1, L_2), the probability that both G_1 and G_2 contain required paths is at least $1/2^r$ when Edge-Disjoint (L_1, L_2)-Paths has a solution.

We can use BFS starting from s_i to determine whether G_i contains an (s_i, t_i)-path of length at most k_i in time $O(m)$, and an algorithm of Fomin et al. [10] to determine whether G_i contains an (s_i, t_i)-path of length exactly l in time $O(2.6181^l m \log^2 n)$. Furthermore, we use a family of (m, r)-universal sets of size $2^r r^{O(\log r)} \log m$ [14] for derandomization. Therefore Edge-Disjoint (L_1, L_2)-Paths can be solved in time

$$2^r r^{O(\log r)} \log m * m = 2^r r^{O(\log r)} m \log n = O(2.01^r m \log n)$$

for $(L_1, L_2) = (\leq k_1, \leq k_2)$, and time

$$2^r r^{O(\log r)} \log m * (2.6181^{k_1} + 2.6181^{k_2}) m \log^2 n = O(5.24^r m \log^3 n)$$

for $(L_1, L_2) = (\leq k_1, = k_2)$ or $(= k_1, = k_2)$. □

For other cases of (L_1, L_2), a random edge partition of G does not, unfortunately, gurantee condition (1) because of the possible existence of a long path in a solution. To handle such cases, we will compute some special edges and then use random partition on such edges to ensure condition (1). For this purpose, we call a vertex v a *nearby-vertex* if $d(s_1, v) + d(v, t_1) \leq k_1$, and call an edge a *nearby-edge* if its two endpoints are both nearby-vertices. We will show that there exists a solution where the number of nearby-edges is bounded above by a polynomial in k_1 and k_2 alone, which enables us to apply random partition to nearby-edges to ensure condition (1) and hence to obtain FPT algorithms. We note that such a clever way of applying random partition has been used by Cygan et al. [6] in obtaining an Eulerian graph by deleting at most k edges.

In the next two subsections, we rely on random partition of nearby-edges to obtain FPT algorithms to solve Edge-Disjoint (L_1, L_2)-Paths for the following four cases of (L_1, L_2): $(\leq k_1, \leq \infty), (= k_1, \leq \infty), (\leq k_1, \geq k_2)$ and $(= k_1, \geq k_2)$.

2.1 One Short and One Unconstrained

In this subsection, we use random partition on nearby-edges to obtain FPT algorithms for Edge-Disjoint (L_1, L_2)-Paths when (L_1, L_2) is $(\leq k_1, \leq \infty)$ or $(= k_1, \leq \infty)$. To lay the foundation of our FPT algorithms, we first present the following crucial property on the number of nearby-edges in a special solution. Recall that a nearby-vertex v satisfies $d(s_1, v) + d(v, t_1) \leq k_1$ and both endpoints of a nearby-edge are nearby-vertices.

Lemma 1. *Let (s_1, t_1) and (s_2, t_2) be two pairs of vertices in a graph $G = (V, E), P_1$ an (s_1, t_1)-path of length at most k_1, and P_2 a minimum-length (s_2, t_2)-path edge-disjoint from P_1. Then*

1. all edges in P_1 are nearby-edges, and
2. P_2 contains at most $(k_1+1)^2$ nearby-edges.

Proof. Statement 1 is obvious and we focus on Statement 2.

For a vertex v in P_2, we say that v is a P_1-*near vertex* if there is a vertex u in P_1 such that G contains a (u, v)-path of length at most $k_1/2$ that is edge-disjoint from P_1. We call v a u-*near vertex* when we want to emphasize the endpoint u, and refer to such a (u, v)-path as a P_1-*near* (u, v)-*path*.

Let v^* be a nearby-vertex in P_2. Since $d(s_1, v^*) + d(v^*, t_1) \leq k_1$, there is an (s_1, v^*)-path or a (t_1, v^*)-path of length at most $k_1/2$. As s_1 and t_1 are vertices of P_1, v^* must be a P_1-near vertex. Therefore each nearby-vertex in P_2 is a P_1-near vertex, and we bound the number of P_1-near vertices to prove this lemma.

Suppose to the contrary that P_2 contains at least $(k_1 + 1)^2 + 1$ P_1-near vertices. Then by pigeonhole principle, there exists a vertex u in P_1 that has at least $k_1 + 2$ u-near vertices. Sort these vertices along P_2 from s_2 to t_2. Let v_1 and v_2 be the first and last vertex respectively. Then the (v_1, v_2)-section of P_2 has length at least $k_1 + 1$. Let W be the (v_1, v_2)-walk concatenating the P_1-near (u, v_1)-path and the P_1-near (u, v_2)-path. Then W contains at most k_1 edges and is edge-disjoint from P_1 by the definition of P_1-near path. So we can replace the (v_1, v_2)-section by W to obtain an (s_2, t_2)-walk that contains an (s_2, t_2)-path shorter than P_2, contradicting to the minimality of P_2. Therefore P_2 contains at most $(k_1 + 1)^2$ P_1-near vertices and thus nearby-vertices, which implies that P_2 contains at most $(k_1 + 1)^2$ nearby-edges. □

The above lemma lays the ground for an FPT algorithm based on random partition. Let $\{E_1, E_2\}$ be a random partition of nearby-edges, and construct $G_1 = G[E_1]$ and $G_2 = G - E(G_1)$. Note that whenever G admits a solution, it has a solution (P_1, P_2) such that P_2 is a minimum-length (s_2, t_2)-path edge disjoint from P_1. Lemma 1 implies that P_1 is inside G_1 with probability $\geq 1/2^{k_1}$, and P_2 is inside G_2 with probability $\geq 1/2^{(k_1+1)^2}$. This ensures that, with probability $\geq 1/2^{k_1}$, G_1 contains an (s_1, t_1)-path of length at most k_1 and, with probability at least $1/2^{(k_1+1)^2}$, G_2 contains an (s_2, t_2)-path. Therefore with probability $\geq 1/2^{k_1+(k_1+1)^2}$, we will be able to find a solution for G by finding an (s_1, t_1)-path of length at most k_1 in G_1 and an (s_2, t_2)-path in G_2. This paves the way for the following randomized FPT algorithm for Edge-Disjoint $(\leq k_1, \leq \infty)$-Paths. Note that the algorithm also works for Edge-Disjoint $(= k_1, \leq \infty)$-Paths once we change "length $\leq k_1$" to "length k_1" in Step 3.

Algorithm 1.

1. Find all nearby-edges in $O(m)$ time by two rounds of BFS, one from s_1 and the other from t_1.
2. Randomly color each nearby-edge by color 1 or 2 with probability 1/2, and color all remaining edges of G by color 2. Let G_i $(i = 1, 2)$ be the graph consisting of edges of color i.
3. Find an (s_1, t_1)-path P_1 of length $\leq k_1$ in G_1, and an (s_2, t_2)-path P_2 in G_2. Return (P_1, P_2) as a solution if both P_1 and P_2 exist, and return "No" otherwise.

Algorithm 1 solves EDGE-DISJOINT $(\leq k_1, \leq \infty)$-PATHS with probability $\geq 1/2^{k_1+(k_1+1)^2}$ and runs in $O(m)$ time, as the two tasks in Step 3 for G_1 and G_2 also take $O(m)$ time. Let m' be the number of nearby-edges and $r = k_1+(k_1+1)^2$. We can use (m', r)-universal sets to derandomize our algorithm, and obtain a deterministic FPT algorithm running in time

$$2^r r^{O(\log r)} \log n * m' = O(2.01^{k_1^2} m \log n).$$

For EDGE-DISJOINT $(= k_1, \leq \infty)$-PATHS, Step 3 takes more time as it takes $O(2.6181^{k_1} m \log^2 n)$ time to find an (s_1, t_1)-path P_1 of length k_1. Therefore our deterministic FPT algorithm for the problem takes time

$$2^r r^{O(\log r)} \log m' * 2.6181^{k_1} m \log^2 n = O(2.01^{k_1^2} m \log^3 n).$$

Theorem 2. EDGE-DISJOINT $(\leq k_1, \leq \infty)$-PATHS *and* EDGE-DISJOINT $(= k_1, \leq \infty)$-PATHS *can be solved in time* $O(2.01^{k_1^2} m \log n)$ *and* $O(2.01^{k_1^2} m \log^3 n)$ *respectively.*

2.2 One Short and One Long

Now we consider EDGE-DISJOINT (L_1, L_2)-PATHS when (L_1, L_2) is $(\leq k_1, \geq k_2)$ or $(= k_1, \geq k_2)$. The main difficulty lies in the possibility that one path may be long, and we overcome this obstacle by the following lemma similar to Lemma 1 to upper bound the number of nearby-edges in a special solution. Again, the lemma enables us to use random partition on nearby-edges to obtain FPT algorithms for both cases.

For an (s_1, t_1)-path P, a *P-valid (s_2, t_2)-path* is an (s_2, t_2)-path that is edge-disjoint from P and has length at least k_2.

Lemma 2. *Let (s_1, t_1) and (s_2, t_2) be two pairs of vertices in a graph $G = (V, E)$, P an (s_1, t_1)-path of length at most k_1, and Q a P-valid (s_2, t_2)-path of minimum length. Then*

1. *all edges in P are nearby-edges, and*
2. *at most $k_1^2 + 3k_1 + 2k_2$ edges of Q are nearby-edges.*

The proof is omitted due to the space limit, and will appear in the full version of this paper.

The above lemma enables us to obtain a randomized FPT for EDGE-DISJOINT $(\leq k_1, \geq k_2)$ by replacing Step 3 of Algorithm 1 as follows:

Step 3: Find an (s_1, t_1)-path P_1 of length $\leq k_1$ in G_1, and an (s_2, t_2)-path P_2 of length $\geq k_2$ in G_2. Return (P_1, P_2) as a solution if both P_1 and P_2 exist, and return "No" otherwise.

By Lemma 2, the randomized algorithm solves EDGE-DISJOINT $(\leq k_1, \geq k_2)$-PATHS with probability $\geq 1/2^{k_1^2+4k_1+2k_2}$. Since an (s_2, t_2)-path P_2 of length $\geq k_2$ can be found in time $8^{k_2+o(k_2)} m \log^2 n$ [10] as mentioned earlier in the introduction, the two tasks in Step 3 takes $8^{k_2+o(k_2)}m\log^2 n$ time and thus the randomized algorithm runs in the same time. Let m' be the number of nearby-edges and $r = k_1^2 + 4k_1 + 2k_2$. We can use (m', r)-universal sets to derandomize our algorithm, and obtain a deterministic FPT algorithm for Edge-Disjoint $(\leq k_1, \geq k_2)$-Paths running in time

$$2^r r^{O(\log r)} \log m' * 8^{k_2+o(k_2)} m \log^2 n = O(2.01^{k_1^2+5k_2} m \log^3 n).$$

For Edge-Disjoint $(= k_1, \geq k_2)$-Paths, Step 3 needs to find an (s_1, t_1)-path P_1 of length k_1 which takes $O(2.6181^{k_1} m \log^2 n)$ time. Therefore our deterministic FPT algorithm for the problem takes time

$$2^r r^{O(\log r)} \log m' * O(2.6181^{k_1} m \log^2 n + 8^{k_2+o(k_2)} m \log^2 n) = O(2.01^{k_1^2+5k_2} m \log^3 n).$$

Theorem 3. *Both* Edge-Disjoint $(\leq k_1, \geq k_2)$-Paths *and* Edge-Disjoint $(= k_1, \geq k_2)$-Paths *can be solved in time* $O(2.01^{k_1^2+5k_2} m \log^3 n)$.

3 Incompressibility

Having obtained FPT algorithms, we are impelled to investigate the existence of polynomial kernels for Edge-Disjoint (L_1, L_2)-Paths. Our findings are negative as we will show that, unless $NP \subseteq coNP/poly$, the problem admits no polynomial kernel for all nine different cases of length constraints (L_1, L_2).

We start with relaxed-composition algorithms defined by Cai and Cai [5], which is a relaxation of composition algorithms introduced by Bodlaender et al. [2] in their pioneer work on the nonexistence of polynomial kernels, and a clipped version of cross-composition [3] without polynomial equivalence relations.

Definition 1 (relaxed-composition [5]). A relaxed-composition algorithm for a parameterized problem Π takes w instances $(I_1, k), \ldots, (I_w, k) \in \Pi$ as input and, in time polynomial in $\sum_{i=1}^{w} |I_i| + k$, outputs an instance $(I, k) \in \Pi$ such that

1. (I, k) is a yes-instance of Π iff some (I_i, k) is a yes-instance of Π, and
2. k' is polynomial in $\max_{i=1}^{w} |I_i| + \log w$.

Note that relaxed-composition algorithms relax the requirement in composition algorithms [2] for parameter k' from polynomial in k to polynomial in $\max_{i=1}^{w} |I_i| + \log w$. As observed by Cai and Cai [5], the following important result is implicitly established in Bodlaender et al. [2].

Theorem 4 [2,3,11]**.** *If an NP-complete parameterized problem admits a relaxed-composition algorithm, then it has no polynomial kernel, unless* $NP \subseteq coNP/poly$.

We also need the following polynomial parameter transformation (ppt-reduction in short).

Definition 2 (ppt-reduction [4,5]**).** A ppt-reduction from a parameterized problem Π to another parameterized problem Π' is an algorithm that, for input $(I, k) \in \Pi$, takes time polynomial in $|I| + k$ and outputs an instance $(I', k) \in \Pi'$ such that

1. (I, k) is a yes-instance of Π iff (I', k') is a yes-instance of Π', and
2. parameter k' is bounded above by a polynomial of k.

Theorem 5 [4]**.** *If there is a ppt-reduction from a parameterized problem* Π *to another parameterized problem* Π'*, then* Π' *admits no polynomial kernel whenever* Π *admits no polynomial kernel.*

Now we show the nonexistence of polynomial kernels for seven easy cases. We first use relaxed-compositions to show the nonexistence of polynomial kernels of (s,t)-k-Path (resp., Long (s,t)-Path) that are NP-complete problems of finding an (s,t)-path of length k (resp., $\geq k$). Then we present ppt-reductions from these two problems to Edge-Disjoint (L_1, L_2)-Paths problems.

Lemma 3. *Both* (s,t)-k-Path *and* Long (s,t)-Path *admit no polynomial kernel unless NP* $\subseteq$ *coNP/poly.*

Proof. Given w instances of (s,t)-k-Path with s_i and t_i being the two terminal vertices of the i-th instance for $1 \leq i \leq w$, we can relaxed-composite these w instances into one instance by identifying s_i (resp., t_i) as one vertex for all $1 \leq i \leq w$. Then, by Theorem 4, (s,t)-k-Path admits no polynomial kernel unless $NP \subseteq coNP/poly$. By the same relaxed-composition, we can deduce that Long (s,t)-Path admits no polynomial kernel unless $NP \subseteq coNP/poly$. □

Theorem 6. Edge-Disjoint (L_1, L_2)-Paths *for* (L_1, L_2) *being* $(\leq k_1, = k_2), (\leq k_1, \geq k_2), (= k_1, = k_2), (= k_1, \leq \infty), (= k_1, \geq k_2), (\geq k_1, \leq \infty)$ *or* $(\geq k_1, \geq k_2)$*, admits no polynomial kernel unless* $NP \subseteq coNP/poly$*.*

Proof. Given an instance of (s,t)-k-Path, we construct an instance of Edge-Disjoint $(= k_1, \leq \infty)$-Paths as following:

1. Set $s_1 = s$ and $t_1 = t$, and $k_1 = k$,
2. add new vertices s_2 and t_2, and edge s_2t_2.

The above reduction is clearly a ppt-reduction, and thus Edge-Disjoint $(= k_1, \leq \infty)$-Paths admits no polynomial kernel unless $NP \subseteq coNP/poly$. For the other six cases, similar ppt-reductions from (s,t)-k-Path or Long (s,t)-Path will work. □

Now we consider the remaining two cases of length constraints $(\leq k_1, \leq k_2)$ and $(\leq k_1, \leq \infty)$. Following our argument for the other cases, we can easily construct ppt-reductions from the problem of determining whether G contains an (s,t)-path of length at most k. Unfortunately, this short path problem is solvable in polynomial time and thus admits a polynomial kernel, which makes such ppt-reductions meaningless for the purpose of proving the nonexistence of polynomial kernels. In fact, these two cases are difficult to deal with, and we will design delicate relaxed-composition algorithms to establish the nonexistence of their polynomial kernels.

Theorem 7. *Both* Edge-Disjoint $(\leq k_1, \leq k_2)$-Paths *and* Edge-Disjoint $(\leq k_1, \leq \infty)$-Paths *admit no polynomial kernel unless* $NP \subseteq coNP/poly$.

Proof. Let $(G^1, \leq k_1, \leq k_2), \ldots, (G^w, \leq k_1, \leq k_2)$ be w instances of Edge-Disjoint $(\leq k_1, \leq k_2)$-Paths, and $n = \max_{i=1}^{w} |V(G_i)|$. Let (s_1^i, t_1^i) and (s_2^i, t_2^i) be the two pairs of vertices of the i-th instance for $1 \leq i \leq w$. Assume that w is a power of two, say $w = 2^d$. Otherwise we can add some redundant no-instances to make w a power of two.

We first show how to composite two instances into one instance, which is the crucial step of our relaxed-composition. Given the i-th instance and j-th instance, we construct a new instance $(G', \leq k_1', \leq k_2')$ as following (See Fig. 1 for an illustration.):

1. Create two pairs of vertices (s_1', t_1') and (s_2', t_2'), and four vertices u_1, u_2, v_1 and v_2.
2. Connect these new vertices with graph G^i and G^j as showed in Fig. 1, where each dashed/dotted edge is a *short-path* of length one, and each normal edge is a *long-path* of length $k_1 + 4$.
3. Denote by G' the new graph and set $k_1' = k_1 + 4, k_2' = k_2 + 3(k_1 + 4) + 1$.

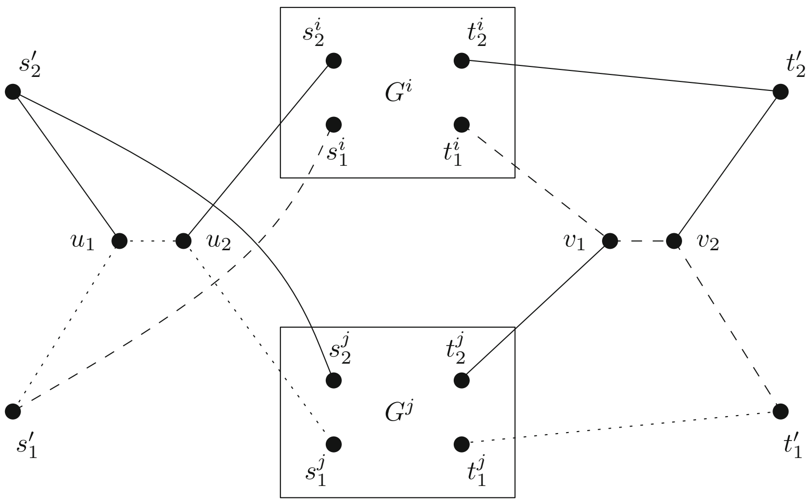

Fig. 1. The relaxed-composition for two instances. Here a dashed/dotted edge is a short-path of length one, and a normal edge is a long-path of length $k_1' = k_1 + 4$.

We claim that $(G', \leq k'_1, \leq k'_2)$ is a yes-instance iff one of these two instances is a yes-instance.

Suppose that one of these two instances has a solution. Without loss of generality, assume that $(G^i, \leq k_1, \leq k_2)$ has a solution (P_1, P_2). Let P'_1 be the (s'_1, t'_1)-path concatenated by P_1 and the four dashed short-paths, and P'_2 be the (s'_2, t'_2)-path going through u_1, u_2, s^i_2 and t^i_2, whose (s^i_2, t^i_2)-section is P_2. By the edge-disjointness between P_1 and P_2, P'_1 and P'_2 are edge-disjoint. Furthermore, we have $|P'_1| \leq k'_1$ and $|P'_2| \leq k'_2$ as $|P'_1| \leq k_1$ and $|P'_2| \leq k_2$. Then (P'_1, P'_2) is a solution of $(G', \leq k'_1, \leq k'_2)$.

Conversely, suppose that (P'_1, P'_2) is a solution of $(G', \leq k'_1, \leq k'_2)$. Since P'_1 has length at most $k'_1 = k_1+4$, and each long-path has length k_1+4, P'_1 contains either all dotted short-paths or dashed short-paths. Assume that P'_1 contains all dotted short-paths. (The argument is similar when P'_1 contains all dashed short-paths.) Then the (s^j_1, t^j_1)-section P_1 of P'_1 is an (s^j_1, t^j_1)-path in G^j of length at most k_1. Moreover, P'_2 must be an (s'_2, t'_2)-path going through the (s'_2, s^j_2)-long-path P_s and the (t^j_2, v_1)-long-path P_t. Since $d(v_1, t'_2) = k_1+5$, the (s^j_2, t^j_2)-section $P_2 \in G^j$ of P'_2 has length at most

$$|P'_2| - |P_s| - |P_t| - d(v_1, t_2) \leq (k_2 + 3k_1 + 13) - 2(k_1 + 4) - (k_1 + 5) \leq k_2.$$

Then (P_1, P_2) is a solution of $(G^j, \leq k_1, \leq k_2)$.

Now we are ready to present our relaxed-composition that contains $d = \log w$ iterations. In the i-th iteration, there are 2^{d-i+1} instances and we group these instances into 2^{d-i} pairs for $1 \leq i \leq d$. For each pair, we composite them into one instance as presented above. Finally, there remains only one instance which completes the relaxed-composition. Let $(\leq k^i_1, \leq k^i_2)$ be the length constraints after the i-th iteration for $0 \leq i \leq d$. Note that $k^0_1 = k_1$ and $k^0_2 = k_2$. The recursion relation for k^i_1 and k^i_2 is

$$k^{i+1}_1 = k^i_1 + 4 \,\text{and}\, k^{i+1}_2 = k^i_2 + 3k^{i+1}_1 + 1,$$

as short-path and long-path respectively have length 1 and k^{i+1}_1 in the i-th iteration. We have $k^i_1 = k_1 + 4i$ and $k^i_2 = k_2 + (3k_1+1)i + 6i(i+1)$ for $0 \leq i \leq d$.

Let $(G'', \leq k''_1, \leq k''_2)$ be the final instance, where $k''_1 = k^d_1 = k_1 + 4d$ and $k''_2 = k^d_1 = k_2 + (3k_1 + 1)d + 6d(d + 1)$. By above proof for the composition of two instances, we can deduce that $(G'', \leq k''_1, \leq k''_2)$ has a solution iff one of these w instances has a solution. Both k''_1 and k''_2 are polynomially bounded in $n + \log w$ as $d = \log w$. This composition is a valid relaxed-composition. Since Edge-Disjoint $(\leq k_1, \leq k_2)$-Paths is NP-complete, by Theorem 4, it admits no polynomial kernel unless $NP \subseteq coNP/poly$.

The relaxed-composition also holds if we discard the length constraint for the second path, i.e. discard the length constraints "$\leq k_2$" and "$\leq k'_2$", which yields that Edge-Disjoint $(\leq k_1, \leq \infty)$-Paths admits no polynomial kernel unless $NP \subseteq coNP/poly$. □

4 Concluding Remarks

We have obtained FPT algorithms to solve Edge-Disjoint (L_1, L_2)-Paths for seven of the nine different cases of length constraints (L_1, L_2), and also established the nonexistence of polynomial kernels for all nine cases, assuming $NP \not\subseteq coNP/poly$. However parameterized complexities of the remaining two cases are open.

Problem 1. Determine the parameterized complexities of Edge-Disjoint $(\geq k_1, \leq \infty)$-Paths and Edge-Disjoint $(\geq k_1, \geq k_2)$-Paths.

Note that an FPT algorithm for Edge-Disjoint $(\geq k_1, \geq k_2)$-Paths will yield a new polynomial-time algorithm to solve Edge-Disjoint Paths for two pairs of terminal vertices (i.e., Edge-Disjoint $(\leq \infty, \leq \infty)$-Paths).

We can consider vertex-disjoint paths, instead of edge-disjoint paths, and form Vertex-Disjoint (L_1, L_2)-Paths problems for nine different length constraints (L_1, L_2). It is straightforward to obtain FPT algorithms by color-coding or random partition for the three cases of (L_1, L_2) being $(\leq k_1, \leq k_2), (= k_1, \leq k_2)$ or $(= k_1, = k_2)$. Interestingly, both Vertex-Disjoint $(\geq k_1, \leq \infty)$-Paths and Vertex-Disjoint $(\geq k_1, \geq k_2)$-Paths can be solved by finding a minor that is a disjoint union of two paths, and thus are FPT by the graph minor theorem. (Note that we can not use this approach to solve Problem 1 by transforming edge-disjoint paths into vertex-disjoint paths through line graphs, because paths in line graphs may not correspond to paths in original graphs.) The remaining four cases seem much harder than their corresponding edge-disjoint counterparts. We note that structural properties similar to Lemmas 1 and 2 seem not hold for vertex-disjoint paths with length constraints.

On the other hand, our proofs for the nonexistence of polynomial kernels also work for Vertex-Disjoint (L_1, L_2)-Paths, and hence Vertex-Disjoint (L_1, L_2)-Paths admits no polynomial kernel unless $NP \subseteq coNP/poly$ for all nine different cases of length constraints (L_1, L_2).

Finally, we can consider both edge-disjoint and vertex-disjoint paths with length constraints for digraphs, which appear to be much harder than these problems on undirected graphs.

Problem 2. For digraphs, determine the parameterized complexity of Edge-Disjoint (L_1, L_2)-Paths and Vertex-Disjoint (L_1, L_2)-Paths for various length constraints (L_1, L_2).

References

1. Bodlaender, H.L.: On linear time minor tests with depth-first search. J. Algorithms **14**(1), 1–23 (1993)
2. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. J. Comput. Syst. Sci. **75**(8), 423–434 (2009)
3. Bodlaender, H.L., Jansen, B.M., Kratsch, S.: Kernelization lower bounds by cross-composition. SIAM J. Discrete Math. **28**(1), 277–305 (2014)

4. Bodlaender, H.L., Thomassé, S., Yeo, A.: Kernel bounds for disjoint cycles and disjoint paths. Theoret. Comput. Sci. **412**, 4570–4578 (2011)
5. Cai, L., Cai, Y.: Incompressibility of H-free edge modification problems. Algorithmica **71**(3), 731–757 (2014)
6. Cygan, M., Marx, D., Pilipczuk, M., Pilipczuk, M., Schlotter, I.: Parameterized complexity of Eulerian deletion problems. Algorithmica **68**(1), 41–61 (2014)
7. Eilam-Tzoreff, T.: The disjoint shortest paths problem. Discrete Appl. Math. **85**(2), 113–138 (1998)
8. Even, S., Itai, A., Shamir, A.: On the complexity of timetable and multicommodity flow problems. SIAM J. Comput. **5**(4), 691–703 (1976)
9. Fomin, F.V., Lokshtanov, D., Panolan, F., Saurabh, S.: Representative sets of product families. In: Schulz, A.S., Wagner, D. (eds.) ESA 2014. LNCS, vol. 8737, pp. 443–454. Springer, Heidelberg (2014)
10. Fomin, F.V., Lokshtanov, D., Saurabh, S.: Efficient computation of representative sets with applications in parameterized and exact algorithms. In: Proceedings of the 25th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 142–151. SIAM (2014)
11. Fortnow, L., Santhanam, R.: Infeasibility of instance compression and succinct PCPs for NP. J. Comput. Syst. Sci. **77**(1), 91–106 (2011)
12. Gabow, H.N., Nie, S.: Finding long paths, cycles and circuits. In: Hong, S.-H., Nagamochi, H., Fukunaga, T. (eds.) ISAAC 2008. LNCS, vol. 5369, pp. 752–763. Springer, Heidelberg (2008)
13. Golovach, P.A., Thilikos, D.M.: Paths of bounded length and their cuts: parameterized complexity and algorithms. Discrete Optim. **8**(1), 72–86 (2011)
14. Naor, M., Schulman, L.J., Srinivasan, A.: Splitters and near-optimal derandomization. In: Proceedings of the 36th Annual Symposium on Foundations of Computer Science, pp. 182–191. IEEE (1995)
15. Ohtsuki, T.: The two disjoint path problem and wire routing design. In: Saito, N., Nishizeki, T. (eds.) Graph Theory and Algorithms. LNCS, vol. 108, pp. 207–216. Springer, Heidelberg (1980)
16. Orlin, J.B.: Max flows in O(nm) time, or better. In: Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing, pp. 765–774. ACM (2013)
17. Robertson, N., Seymour, P.D.: Graph minors. XIII. The disjoint paths problem. J. Comb. Theory Ser. B **63**(1), 65–110 (1995)
18. Seymour, P.D.: Disjoint paths in graphs. Discrete Math. **29**(3), 293–309 (1980)
19. Shachnai, H., Zehavi, M.: Representative families: a unified tradeoff-based approach. J. Comput. Syst. Sci. **82**(3), 488–502 (2016)
20. Shiloach, Y.: A polynomial solution to the undirected two paths problem. J. ACM **27**(3), 445–456 (1980)
21. Thomassen, C.: 2-linked graphs. Eur. J. Comb. **1**(4), 371–378 (1980)
22. Tragoudas, S., Varol, Y.L.: Computing disjoint paths with length constraints. In: d'Amore, F., Franciosa, P.G., Marchetti-Spaccamela, A. (eds.) Graph-Theoretic Concepts in Computer Science. LNCS, vol. 1197, pp. 375–389. Springer, Heidelberg (1997)
23. Zehavi, M.: Mixing color coding-related techniques. In: Bansal, N., Finocchi, I. (eds.) ESA 2015. LNCS, vol. 9294, pp. 1037–1049. Springer, Heidelberg (2015)