Tight Approximation and Kernelization Bounds for Vertex-Disjoint Shortest Paths

Matthias Bentert
University of Bergen, Norway

Fedor V. Fomin
University of Bergen, Norway

Petr A. Golovach
University of Bergen, Norway

Abstract

We examine the possibility of approximating Maximum Vertex-Disjoint Shortest Paths. In this problem, the input is an edge-weighted (directed or undirected) n-vertex graph G along with k terminal pairs $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$. The task is to connect as many terminal pairs as possible by pairwise vertex-disjoint paths such that each path is a shortest path between the respective terminals. Our work is anchored in the recent breakthrough by Lochet [SODA '21], which demonstrates the polynomial-time solvability of the problem for a fixed value of k.

Lochet's result implies the existence of a polynomial-time ck-approximation for Maximum Vertex-Disjoint Shortest Paths, where $c \leq 1$ is a constant. (One can guess $1/c$ terminal pairs to connect in $k^{O(1/c)}$ time and then utilize Lochet's algorithm to compute the solution in $n^{f(1/c)}$ time.) Our first result suggests that this approximation algorithm is, in a sense, the best we can hope for. More precisely, assuming the gap-ETH, we exclude the existence of an $o(k)$-approximation within $f(k) \cdot \text{poly}(n)$ time for any function f that only depends on k.

Our second result demonstrates the infeasibility of achieving an approximation ratio of $m^{1/2-\varepsilon}$ in polynomial time, unless P = NP. It is not difficult to show that a greedy algorithm selecting a path with the minimum number of arcs results in a $\lceil\sqrt{\ell}\rceil$-approximation, where ℓ is the number of edges in all the paths of an optimal solution. Since $\ell \leq n$, this underscores the tightness of the $m^{1/2-\varepsilon}$-inapproximability bound.

Additionally, we establish that Maximum Vertex-Disjoint Shortest Paths is fixed-parameter tractable when parameterized by ℓ but does not admit a polynomial kernel. Our hardness results hold for undirected graphs with unit weights, while our positive results extend to scenarios where the input graph is directed and features arbitrary (non-negative) edge weights.

2012 ACM Subject Classification Theory of computation → Approximation algorithms analysis; Theory of computation → Shortest paths; Theory of computation → Problems, reductions and completeness; Theory of computation → Fixed parameter tractability

Keywords and phrases Inapproximability, Fixed-parameter tractability, Parameterized approximation

Digital Object Identifier 10.4230/LIPIcs.STACS.2025.17

1 Introduction

We study a variant of the well-known problem Vertex-Disjoint Paths. In the latter, the input comprises a (directed or undirected) graph G and k terminal pairs. The task is to identify whether pairwise vertex-disjoint paths can connect all terminals. Vertex-Disjoint Paths has long been established as NP-complete [21] and has played a pivotal role in the graph-minor project by Robertson and Seymour [29].

Eilam-Tzoreff [14] introduced a variant of Vertex-Disjoint Paths where all paths in the solution must be *shortest* paths between the respective terminals. The parameterized complexity of this variant, known as Vertex-Disjoint Shortest Paths, was recently

42nd International Symposium on Theoretical Aspects of Computer Science (STACS 2025).
Editors: Olaf Beyersdorff, Michał Pilipczuk, Elaine Pimentel, and Nguyễn Kim Thắng;
Article No. 17; pp. 17:1–17:17
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

resolved [25] and subsequently the running time improved [3]: The problem, parameterized by k, is W[1]-hard and in XP (that is, polynomial-time solvable for constant k) for undirected graphs. On directed graphs, the problem is NP-hard already for $k = 2$ if zero-weight edges are allowed [16]. The problem is solvable in polynomial time for $k = 2$ for positive edge weights [4]. It is NP-hard when k is part of the input and the complexity for constant $k > 2$ remains open.

The optimization variant of Vertex-Disjoint Shortest Paths, where not necessarily all terminal pairs need to be connected, but at least p of them, is referred to as Maximum Vertex-Disjoint Shortest Paths.

Maximum Vertex-Disjoint Shortest Paths

Input: A graph $G = (V, E)$, an edge-length function $w\colon E \to \mathbb{Q}_{\geq 0}$, terminal pairs $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$ where $s_i \neq t_i$ for $i \in [k]$, and an integer p.

Question: Is there a set $S \subseteq [k]$ with $|S| \geq p$ such that there is a collection $\mathcal{C} = \{P_i\}_{i \in S}$ of pairwise vertex-disjoint paths such that for each $i \in S$, path P_i is a shortest path from s_i to t_i?

A few remarks are in order. In the literature concerning Vertex-Disjoint Paths and its variants, one usually distinguishes between vertex-disjoint and internally vertex-disjoint paths. In the latter, two paths in a solution might share common endpoints while in the former, all paths must be completely vertex disjoint – including the two ends. We focus on the variant where paths must be completely vertex-disjoint, but most of our results also hold for internally vertex-disjoint paths.

Note that Vertex-Disjoint Shortest Paths is a special case of Maximum Vertex-Disjoint Shortest Paths with $p = k$. For the maximization version, we are not given p as input but are instead asked to find a set S that is as large as possible. Slightly abusing notation, we do not distinguish between these two variants and refer to both as Maximum Vertex-Disjoint Shortest Paths.

While parameterization by k yields strong hardness bounds (both in terms of parameterized complexity and, as we will show later, approximation), another natural parameterization would be the sum of path lengths in a solution. We initiate the study of a related parameter ℓ, the minimum number of edges in an optimal solution (assuming the instance is a yes-instance, otherwise, we define $\ell = n$). If we confine all edge weights to be positive integers, then ℓ serves as a lower bound for the sum of path lengths. Since our hardness results apply to unweighted graphs, studying ℓ instead of the sum of path lengths does not weaken the negative results and ℓ proves to be very useful for approximation and parameterized algorithms. Note that the sum of path lengths is not a suitable parameter as dividing all edge weights by $m \cdot w_{\max}$ (where $w_{\max}$ is the maximum weight of any edge in the input) yields an equivalent instance where the sum of path lengths in the solution is at most one.

For the parameterized complexity of Maximum Vertex-Disjoint Shortest Paths, we note that the results for Vertex-Disjoint Shortest Paths [3, 25] for the parameterization by k directly translate for Maximum Vertex-Disjoint Shortest Paths parameterized by p. The problem is W[1]-hard as a generalization of Vertex-Disjoint Shortest Paths, and to obtain an XP algorithm, it is sufficient to observe that in $n^{O(p)}$ time we can guess a set $S \subseteq [k]$ of size p and apply the XP algorithm for Vertex-Disjoint Shortest Paths for the selected set of terminal pairs.

Before the recent work of Chitnis, Thomas, and Wirth [8], little was known about approximation algorithms for the Maximum Vertex-Disjoint Shortest Paths problem. Chitnis, Thomas, and Wirth demonstrated that no $(2 - \varepsilon)$-approximation could be achieved in time $f(k) \cdot n^{o(k)}$ assuming the gap-ETH.

Table 1 Overview of our results. New results are bold. All hardness results hold for unweighted and undirected graphs, while all new algorithmic results hold even for directed graphs with arbitrary non-negative edge weights.

Parameter	Exact	Approximation
no	NP-complete	**no $\mathbf{m^{1/2-\varepsilon}}$-approximation in poly(n) time**
k	XP and W[1]-hard	**no o(k)-approximation in $\mathbf{f(k) \cdot poly(n)}$ time**
ℓ	**FPT** and **no poly kernel**	**$\lceil\sqrt{\ell}\rceil$-approximation**

For the related Maximum Vertex-Disjoint Paths, where the task is to connect the maximum number of terminal pairs by disjoint but not necessarily shortest paths, $O(\sqrt{n})$-approximation algorithms are due to Kleinberg [23] and Kolliopoulos and Stein [24]. The best known lower bounds for this variant are $2^{\Omega(\sqrt{\log n})}$ and $n^{\Omega(1/(\log\log n)^2)}$. The first lower bound holds even if the input graph is an unweighted planar graph, while the second holds even if the input graph is an unweighted grid graph [9, 10]. For these two special cases, there are approximation algorithms achieving ratios $\tilde{O}(n^{9/19})$ and $\tilde{O}(n^{1/4})$, respectively [9, 10].

When requiring the solution paths to be edge-disjoint rather than vertex-disjoint, it is known that even computing a $m^{1/2-\varepsilon}$-approximation is NP-hard in the directed setting [18]. There have also been some studies on relaxing the notion so that each edge appears in at most $c > 1$ of the solution paths. The integer c is called the congestion and the currently best known approximation algorithm achieves a poly$(\log n)$-approximation with $c = 2$ [11].

Our results. We show that computing a $m^{1/2-\varepsilon}$-approximation for Maximum Vertex-Disjoint Shortest Paths is NP-hard for any $\varepsilon > 0$ (Theorem 3). Moreover in terms of FPT-approximations, we demonstrate in Theorem 1 that any $k^{o(1)}$-approximation in time $f(k) \cdot \text{poly}(n)$ implies FPT = W[1] and that it is impossible to achieve an $o(k)$-approximation in time $f(k) \cdot \text{poly}(n)$ unless the gap-ETH fails. This significantly improves the current state of approximation results for Maximum Vertex-Disjoint Shortest Paths in two ways. First, we use the weaker assumption FPT $\neq$ W[1] instead of the gap-ETH. Second, our theorem excludes approximation factors polynomial in the input size, rather than only constant factors larger than 2 as shown by Chitnis et al. [8].

We complement the first lower bound by showing that a simple greedy strategy for Maximum Vertex-Disjoint Paths achieves a $\lceil\sqrt{\ell}\rceil$-approximation also for Maximum Vertex-Disjoint Shortest Paths (Theorem 4). In Theorems 5 and 7, we show that Maximum Vertex-Disjoint Shortest Paths is fixed-parameter tractable when parameterized by ℓ, but it does not admit a polynomial kernel. We mention that our hardness results hold for undirected graphs with unit weights, and all our positive results hold even for directed and edge-weighted input graphs. We summarize our results in Table 1.

2 Preliminaries

For a positive integer x, we denote by $[x] = \{1, 2, \dots, x\}$ the set of all positive integers at most x. We denote by $G = (V, E)$ a graph and by n and m the number of vertices and edges in G, respectively. A graph G is said to be *k-partite* if V can be partitioned into k disjoint sets $V_1, V_2, \dots, V_k$ such that each set V_i induces an independent set, that is, there is no edge $\{u, v\} \in E$ with $\{u, v\} \subseteq V_i$ for some $i \in [k]$. The *degree* of a vertex v is the number of edges in E that contain v as an endpoint and the *maximum degree* of a graph is the highest degree of any vertex in the graph.

A *path* in a graph G is a sequence $(v_0, v_1, \dots, v_\ell)$ of distinct vertices such that each pair (v_{i-1}, v_i) is connected by an edge in G. The first and last vertex v_0 and v_ℓ are called the *ends* of P. We also say that P is a path *from* v_0 *to* v_ℓ or a v_0-v_ℓ-path. The length of a path is the sum of its edge lengths or simply the number ℓ of edges if the graph is unweighted. For two vertices v, w, we denote the length of a shortest v-w-path in G by $\mathrm{dist}_G(v, w)$ or $\mathrm{dist}(v, w)$ if the graph G is clear from the context.

We assume the reader to be familiar with the big-O notation and basic concepts in computational complexity like NP-completeness and reductions. We refer to the textbook by Garey and Johnson [17] for an introduction.

For a detailed introduction to parameterized complexity and kernelization, we refer the reader to the text books by Cygan et al. [12] and Fomin et al. [15]. A *parameterized problem* P is a language containing pairs (I, ρ) where I is an instance of an (unparameterized) problem and ρ is an integer called the *parameter*. In this paper, the parameter will usually be either the number k of terminal pairs or the minimum number ℓ of edges in a solution (a maximum collection of vertex-disjoint shortest paths between terminal pairs). A parameterized problem P is *fixed-parameter tractable* if there exists an algorithm solving any instance (I, ρ) of P in $f(\rho) \cdot \mathrm{poly}(|I|)$ time, where f is some computable function only depending on ρ. To show that a problem is presumably not fixed-parameter tractable, one usually shows that the problem is hard for a complexity class known as W[1]. The class *XP* contains all parameterized problems which can be solved in $|I|^{f(\rho)}$ time, that is, in polynomial time if ρ is constant. A parameterized problem is said to admit a *polynomial kernel*, if there is a polynomial-time algorithm that given an instance (I, ρ) computes an equivalent instance (I', ρ') (called the *kernel*) such that $|I'| + \rho'$ are upper-bounded by a polynomial in ρ. It is known that any parameterized problem admitting a polynomial kernel is fixed-parameter tractable and each fixed-parameter tractable problem is contained in XP.

An α-approximation algorithm for a maximization problem is a polynomial-time algorithm that for any input returns a solution of size at least $^{\text{OPT}}/_{\alpha}$ where OPT is the size of an optimal solution. A parameterized α-approximation algorithm also returns a solution of size at least $^{\text{OPT}}/_{\alpha}$, but its running time is allowed to be $f(\rho) \cdot \mathrm{poly}(n)$, where ρ is the parameter and f is some computable function only depending on ρ. In this work, we always consider (unparameterized) approximation algorithms unless we specifically state a parameterized running time.

To exclude an α-approximation for an optimization problem, one can use the framework of *approximation-preserving reductions*. A strict approximation-preserving reduction is a pair of algorithms – called the *reduction algorithm* and the *solution-lifting algorithm* – that both run in polynomial time and satisfy the following. The reduction algorithm takes as input an instance I of a problem L and produces an instance I' of a problem L'. The solution-lifting algorithm takes any solution S' of I' and transforms it into a solution S of I such that if S' is an α-approximation for I' for some $\alpha \geq 1$, then S is an α-approximation for I. If a strict approximation-preserving reduction from L to L' exists and L is hard to approximate within some value β, then L' is also hard to approximate within β.

The *exponential time hypothesis (ETH)* introduced by Impagliazzo and Paturi [20] states that there is some $\varepsilon > 0$ such that each (unparameterized) algorithm solving 3-Sat takes at least $2^{\varepsilon n + o(n)}$ time, where n is the number of variables in the input instance. A stronger conjecture called the *gap-ETH* was independently introduced by Dinur [13] and Manurangsi and Raghavendra [27]. It states that there exist $\varepsilon, \delta > 0$ such that any $(1 - \varepsilon)$-approximation algorithm for Max 3-Sat[1] takes at least $2^{\delta n + o(n)}$ time.

[1] Max 3-Sat is a generalization of 3-Sat where the question is not whether the input formula is satisfiable

3 Approximation

In this section, we show that Maximum Vertex-Disjoint Shortest Paths admits no $o(k)$-approximation in $f(k) \cdot \text{poly}(n)$ time unless the gap-ETH fails and no $m^{1/2-\varepsilon}$-approximation in polynomial time unless P = NP. We complement the latter result by developing a $\lceil\sqrt{\ell}\rceil$-approximation algorithm that runs in polynomial time. We start with a reduction based on a previous reduction by Bentert et al. [3] and make it approximation-preserving.[2] Moreover, our result is tight in the sense that a k-approximation can be computed in polynomial time by simply connecting any terminal pair by a shortest path. A ck-approximation for any constant $c \leq 1$ can also be computed in polynomial time by guessing $\frac{1}{c}$ terminal pairs to connect and then using the XP-time algorithm by Bentert et al. [3] to find a solution. Note that since c is a constant, the XP-time algorithm for $\frac{1}{c}$ terminal pairs runs in polynomial time.

▶ **Theorem 1.** *Unless FPT = W[1], Maximum Vertex-Disjoint Shortest Paths cannot be $k^{o(1)}$-approximated in $f(k) \cdot \text{poly}(n)$ time, and assuming the gap-ETH, it cannot be $o(k)$-approximated in $f(k) \cdot \text{poly}(n)$ time. All of these results hold even for subcubic graphs with terminals of degree one.*

Proof. We present a strict approximation-preserving reduction from Multicolored Clique to Maximum Vertex-Disjoint Shortest Paths such that the maximum degree is three and each terminal vertex has degree one. Moreover, the maximum number OPT of vertex-disjoint shortest paths between terminal pairs will be equal to the largest clique in the original instance. The theorem then follows from the fact that a $f(k) \cdot \text{poly}(n)$-time $k^{o(1)}$-approximation for Clique would imply that FPT = W[1] [7, 22], a $f(k) \cdot \text{poly}(n)$-time $o(k)$-approximation for Clique refutes the gap-ETH [6], and that the textbook reduction from Clique to Multicolored Clique only increases the number of vertices by a quadratic factor and does not change the size of a largest clique in the graph [12].

The reduction is depicted in Figure 1 and works as follows. Let $G = (V, E)$ be a k-partite graph (or equivalently a graph colored with k colors where all vertices of any color form an independent set) with ν vertices of each color. Let $V_i = \{v^i_1, v^i_2, \ldots, v^i_\nu\}$ be the set of vertices of color $i \in [k]$ in G. We start with a terminal pair (s_i, t_i) for each color i and a pair of (non-terminal) vertices (s^j_i, t^j_i) for each vertex $v^i_j \in V_i$. Next for each color i, we add a binary tree of height $\lceil\log(\nu)\rceil$ where the vertices s^j_i are the leaves for all $v^i_j \in V_i$. We make s_i adjacent to the root of the binary tree. Analogously, we add a binary tree of the same height with leaves t^i_j and make t_i adjacent to the root. Next, we add a crossing gadget for each pair of vertices (v^i_j, v^a_b) with $i < a$. If $\{v^i_j, v^a_b\} \in E$, then the gadget consists of four vertices $u^{i,a}_{j,b}, v^{i,a}_{j,b}, x^{i,a}_{j,b}$, and $y^{i,a}_{j,b}$ and edges $\{u^{i,a}_{j,b}, v^{i,a}_{j,b}\}$ and $\{x^{i,a}_{j,b}, y^{i,a}_{j,b}\}$. If $\{v^i_j, v^a_b\} \notin E$, then the gadget consists of only two vertices $u^{i,a}_{j,b}$ and $v^{i,a}_{j,b}$ and the edge $\{u^{i,a}_{j,b}, v^{i,a}_{j,b}\}$. For the sake of notational convenience, we will in the latter case also denote $u^{i,a}_{j,b}$ by $x^{i,a}_{j,b}$ and $v^{i,a}_{j,b}$ by $y^{i,a}_{j,b}$. To complete the construction, we connect the different gadgets as follows. First, we connect via paths of length two $v^{i,a}_{j,b}$ and $u^{i,a}_{j,b+1}$ for all $b < \nu$ and $y^{i,a}_{j,b}$ and $x^{i,a}_{j-1,b}$ for all $j > 1$. Second, we connect via paths of length two the vertices $v^{i,a}_{j,\nu}$ to $u^{i,a+1}_{j,1}$ for all $j \in [\nu]$ and all $a < k$

but rather how many clauses can be satisfied simultaneously.

[2] We mention in passing that essentially the same modification to the reduction by Bentert et al. has been found by Akmal et al. [1] in independent research. While we use the modification to show stronger inapproximability bounds, they use it to show stronger fine-grained hardness results with respect to the minimum degree of a polynomial-time algorithm for constant k.

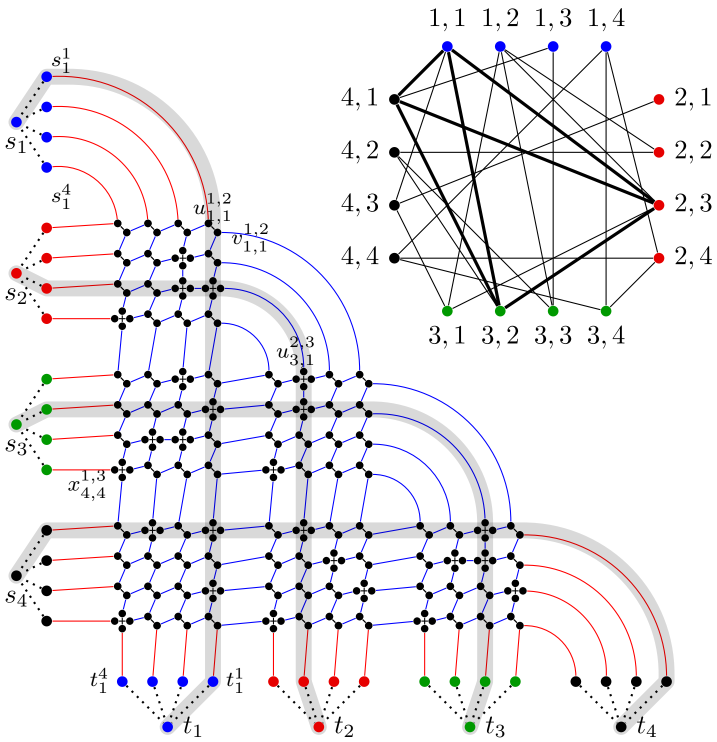

Figure 1 An illustration of the reduction from Multicolored Clique to Maximum Vertex-Disjoint Shortest Paths.
Top right: Example instance for Multicolored Clique with $k = 4$ colors and $n = 4$ vertices per color. A multicolored clique is highlighted (by thick edges).
Bottom left: The constructed instance with the four shortest paths corresponding to the vertices of the clique highlighted. Note that these paths are pairwise disjoint. The dotted edges (incident to s_i and t_i vertices) indicate binary trees (where all leaves have distance $\lceil \log(\nu) \rceil$ from the root). Red edges indicate paths of length 2ν and blue edges indicate paths of length 2.

and $y_{1,b}^{i,a}$ to $x_{\nu,b}^{i+1,a}$ for all $b \in [\nu]$ and all $i < a - 1$. Third, we connect also via paths of length two $y_{1,b}^{i,i+1}$ to $u_{b,1}^{i+1,i+2}$ for all $i < k - 1$ and all $b \in [\nu]$. Next, we connect via paths of length 2ν each vertex s_i^j to $x_{\nu,j}^{1,i}$ for each $i > 1$ and $j \in [\nu]$. Similarly, $v_{j,\nu}^{i,k}$ is connected to t_i^j via paths of length 2ν. Finally, we connect s_j^1 with $u_{j,1}^{1,2}$ for all $j \in [\nu]$ and $y_{1,j}^{k-2,k-1}$ with t_j^k for all $j \in [\nu]$. This concludes the construction.

We next prove that all shortest s_i-t_i-paths are of the form

$$s_i - s_i^j - x_{\nu,j}^{1,i} - y_{1,j}^{i-1,i} - u_{j,1}^{i,i+1} - v_{j,\nu}^{i,k} - t_i^j - t_i \tag{1}$$

for some $j \in [\nu]$ and where the s_1-t_1-paths go directly from s_i^j to $u_{j,1}^{1,2}$ and the s_k-t_k-paths go directly from $y_{1,j}^{k-1,k}$ to t_j^i. We say that the respective path is the j^{th} *canonical path* for color i.

To show the above claim, first note that the distance from s_i to any vertex s_i^j is the same value $x = \lceil \log(\nu) \rceil + 1$ for all pairs of indices i and j. Moreover, the same holds for t_i and t_i^j, each s_i-t_i-path contains at least one vertex s_i^j and one vertex $t_i^{j'}$ for some $j, j' \in [\nu]$, and all paths of the form in Equation (1) are of length $y = 2x + 4\nu + 3(k - 1)\nu - 2$. We first show that each s_i-t_i-path of length at most y contains an edge of the form $y_{1,b}^{i,i+1}$ to $u_{b,1}^{i+1,i+2}$. Consider the graph where all of these edges are removed. Note that due to the grid-like structure, the distance between s_i and $x_{j,b}^{i',a}$ for any values $i' \leq i \leq a, j$, and b is at least $x+2\nu+3(i'-1)\nu+3(\nu-j)$ if $i = a$ and at least $x+2\nu+3(i'-1)\nu+3(a-i)\nu+3(\nu-j)+3b$ if $i < a$.[3] Hence, all shortest s_i-t_i-paths use an edge of the form $y_{1,b}^{i,i+1}$ to $u_{b,1}^{i+1,i+2}$ and the shortest path from s_i^j to some vertex $y_{1,b}^{i,i+1}$ is to the vertex $y_{1,j}^{i,i+1}$. Note that the other endpoint of the specified edge is $u_{j,1}^{i,i+1}$ and the shortest path to t_i now goes via t_i^j for analogous reasons. Thus, all shortest s_i-t_i-paths have the form (1).

We next prove that any set of p disjoint shortest paths between terminal pairs (s_i, t_i) in the constructed graph has a one-to-one correspondence to a multicolored clique of size p for any p. For the first direction, assume that there is a set P of disjoint shortest paths between p terminal pairs. Let $S \subseteq [k]$ be the set of indices such that the paths in P connect s_i and t_i for each $i \in S$. Moreover, let j_i be the index such that the shortest s_i-t_i-path in P is the $j_i{}^{\text{th}}$ canonical path for i for each $i \in S$. Now consider the set $K = \{v_{j_i}^i \mid i \in S\}$ of vertices in G. Clearly K contains at most one vertex of each color and is of size p as S is of size p. It remains to show that K induces a clique in G. To this end, consider any two vertices $v_{j_i}^i, v_{j_{i'}}^{i'} \in K$. We assume without loss of generality that $i < i'$. By assumption, the $j_i{}^{\text{th}}$ canonical path for i and the $j_{i'}{}^{\text{th}}$ canonical path for i' are disjoint. This implies that $u_{j_i,j_{i'}}^{i,i'} \neq x_{j_i,j_{i'}}^{i,i'}$ as the $j_i{}^{\text{th}}$ canonical path for i contains the former and the $j_{i'}{}^{\text{th}}$ canonical path for i' contains the latter. By construction, this means that $\{v_{j_i}^i, v_{j_{i'}}^{i'}\} \in E$. Since the two vertices were chosen arbitrarily, it follows that all vertices in K are pairwise adjacent, that is, K induces a multicolored clique of size p.

For the other direction assume that there is a multicolored clique $C = \{v_{j_1}^{i_1}, v_{j_2}^{i_2}, \ldots, v_{j_p}^{i_p}\}$ of size p in G. We will show that the $j_q{}^{\text{th}}$ canonical path for i_q is vertex-disjoint from the $j_r{}^{\text{th}}$ canonical path for i_r for all $q \neq r \in [p]$. Let q, r be two arbitrary distinct indices in $[p]$ and let without loss of generality be $q < r$. Note that the two mentioned paths can only overlap in vertices $u_{j_q,j_r}^{i_q,i_r}, v_{j_q,j_r}^{i_q,i_r}, x_{j_q,j_r}^{i_q,i_r}$, or $y_{j_q,j_r}^{i_q,i_r}$ and that the $j_q{}^{\text{th}}$ canonical path for i_q only contains vertices $u_{j_q,j_r}^{i_q,i_r}$ and $v_{j_q,j_r}^{i_q,i_r}$ and the $j_r{}^{\text{th}}$ canonical path for i_r only contains $x_{j_q,j_r}^{i_q,i_r}$ and $y_{j_q,j_r}^{i_q,i_r}$. Moreover, since by assumption $v_{j_q}^{i_q}$ and $v_{j_r}^{i_r}$ are adjacent, it holds by construction that $u_{j_q,j_r}^{i_q,i_r}, v_{j_q,j_r}^{i_q,i_r}, x_{j_q,j_r}^{i_q,i_r}$, and $y_{j_q,j_r}^{i_q,i_r}$ are four distinct vertices. Thus, we found vertex disjoint paths between p distinct terminal pairs. This concludes the proof of correctness.

To finish the proof, observe that the constructed instance has maximum degree three, all terminal vertices have degree one, and the construction can be computed in polynomial time. ◀

We mention in passing that in graphs of maximum degree three with terminal vertices of degree one, two paths are vertex disjoint if and only if they are edge disjoint. Hence, Theorem 1 also holds for Maximum Edge-Disjoint Shortest Paths, the edge-disjoint version of Maximum Vertex-Disjoint Shortest Paths.

▶ **Corollary 2.** *Unless FPT = W[1], Maximum Edge-Disjoint Shortest Paths cannot be $k^{o(1)}$-approximated in $f(k) \cdot \text{poly}(n)$ time, and assuming the gap-ETH, it cannot be $o(k)$-approximated in $f(k) \cdot \text{poly}(n)$ time. All of these results hold even for subcubic graphs with terminals of degree one.*

---

[3] We mention that there are some pairs of vertices $x_{j_1,b_1}^{i_1,a_1}$ and $x_{j_2,b_2}^{i_2,a_2}$, where the distance between the two is less than $3(|i_1-i_2|+|a_1-a_2|)\nu+3(|j_1-j_2|+|b_1-b_2|)$. An example is the pair $(x_{1,1}^{1,2} = u_{1,1}^{1,2}, x_{2,2}^{1,2})$ in Figure 1 with a distance of 4. However, in all examples it holds that $i_1\nu - j_1 \neq i_2\nu - j_2$ and $a_1\nu + b_1 \neq a_2\nu + b_2$ such that the left side is either smaller in both inequalities or larger in both inequalities. Hence, these pairs cannot be used as shortcuts as they move "down and left" instead of towards "down and right" in Figure 1.

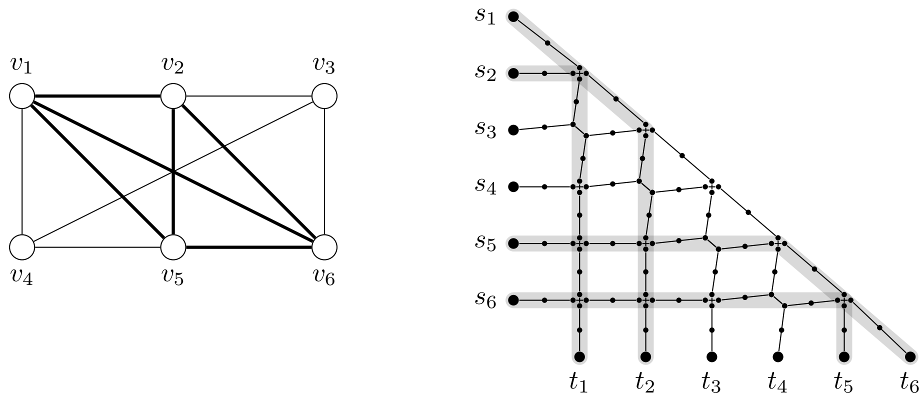

Figure 2 An illustration of the reduction from CLIQUE to MAXIMUM VERTEX-DISJOINT SHORTEST PATHS.
Left side: Example instance for CLIQUE with a highlighted solution (by thick edges).
Right side: The constructed instance with the four shortest paths corresponding to the solution on the left side highlighted. Note that each shortest s_i-t_i-path uses exactly two of the diagonal edges.

We continue with an unparameterized lower bound by establishing that computing a $m^{\frac{1}{2}-\varepsilon}$-approximation is NP-hard. We mention that the reduction is quite similar to the reduction in the proof for Theorem 1.

▶ **Theorem 3.** *Computing a $m^{1/2-\varepsilon}$-approximation for any $\varepsilon > 0$ for MAXIMUM VERTEX-DISJOINT SHORTEST PATHS is NP-hard.*

Proof. It is known that computing a $n^{1-\varepsilon}$-approximation for CLIQUE is NP-hard [19, 30]. We present an approximation-preserving reduction from CLIQUE to MAXIMUM VERTEX-DISJOINT SHORTEST PATHS based on Theorem 1. We use basically the same reduction as in Theorem 1 but we start from an instance of CLIQUE and have a separate terminal pair for each vertex in the graph. Moreover, we do not require the binary trees pending from the terminal vertices and neither do we require long induced paths (red edges in Figure 1). These are instead paths with one internal vertex. An illustration of the modified reduction is given in Figure 2. Note that the number of vertices and edges in the graph is at most $3N^2$, where N is the number of vertices in the instance of CLIQUE. Moreover, for each terminal pair (s_i, t_i), there is exactly one shortest s_i-t_i-path (the path that moves horizontally in Figure 2 until it reaches the main diagonal, then uses exactly two edges on the diagonal, and finally moves vertically to t_i).

We next prove that for any p, there is a one-to-one correspondence between a set of p disjoint shortest paths between terminal pairs (s_i, t_i) in the constructed graph and a clique of size p in the input graph. For the first direction, assume that there is a set P of disjoint shortest paths between p terminal pairs. Let $S \subseteq [k]$ be the set of indices such that the paths in P connect s_i and t_i for each $i \in S$. Now consider the set $K = \{v_i \mid i \in S\}$ of vertices in G. Clearly K contains p vertices. It remains to show that K induces a clique in G. To this end, consider any two vertices $v_i, v_j \in K$. We assume without loss of generality that $i < j$. By assumption, the unique shortest s_i-t_i-path and the unique shortest s_j-t_j-path are vertex-disjoint. By the description of the shortest paths between terminal pairs and the fact that s_i is higher than s_j and t_i is to the left of t_j, it holds that the two considered paths both visit the region that is to the right of s_j and above t_i. This implies that two edges must be crossing at this position, that is, there are four vertices in the described region and not only two. By construction, this means that $\{v_i, v_j\} \in E$. Since the two vertices were chosen arbitrarily, it follows that all vertices in K are pairwise adjacent, that is, K induces a clique of size p in the input graph.

For the other direction assume that there is a clique $C = \{v_{i_1}, v_{i_2}, \ldots, v_{i_p}\}$ of size p in the input graph. We will show that the unique shortest s_{i_q}-t_{i_q}-path is vertex-disjoint from the unique shortest s_{i_r}-t_{i_r}-path for all $q \neq r \in [p]$. Let q, r be two arbitrary distinct indices in $[p]$ and let without loss of generality be $q < r$. Note that the two mentioned paths can only overlap in the region that is to the right of s_{i_r} and above t_{i_q}. Moreover, since by assumption v_{i_q} and v_{i_r} are adjacent, it holds by construction that there are four distinct vertices in the described region and the two described paths are indeed vertex-disjoint. Thus, we found vertex disjoint paths between p distinct terminal pairs.

We conclude by analyzing the approximation ratio. Note that we technically did not prove a strict reduction for the factor $m^{1-\varepsilon}$ as the number of vertices in the original instance and the number of edges in the constructed instance are not identical. Still, the number m of edges in the constructed instance is at most $3N^2$, where N is the number of vertices in the original instance of Clique. Hence, any $m^{1/2-\varepsilon}$-approximation for Maximum Vertex-Disjoint Shortest Paths corresponds to a $(3N^2)^{1/2-\varepsilon} = N^{1-\varepsilon'}$-approximation for Clique for some $0 < \varepsilon' < 2\varepsilon$ and therefore computing a $m^{1/2-\varepsilon}$-approximation for Maximum Vertex-Disjoint Shortest Paths is NP-hard. ◀

Note that the maximum degree of the constructed instance is again three and all terminal vertices are of degree one. Thus, Theorem 3 also holds for the edge disjoint version of Maximum Vertex-Disjoint Shortest Paths. However in this case, a very similar result was already known before. Guruswami et al. [18] showed that computing a $m^{1/2-\varepsilon}$-approximation is NP-hard for a related problem called Length Bounded Edge-Disjoint Paths. Their reduction immediately implies the same hardness for Maximum Edge-Disjoint Shortest Paths. To the best of our knowledge, the best known unparameterized approximation lower bound for Maximum Vertex-Disjoint Shortest Paths was the $2 - \varepsilon$ lower bound due to Chitnis et al. [8] and we are not aware of any lower bound for Maximum Vertex-Disjoint Paths.

We next show that this result is tight, that is, we show how to compute a $\sqrt{n}$-approximation for Maximum Vertex-Disjoint Shortest Paths in polynomial time. We also show that the same algorithm achieves a $\lceil\sqrt{\ell}\rceil$-approximation. Note that we can always assume that $\ell \leq n$ as a set of vertex-disjoint paths is a forest and the number of edges in a forest is less than its number of vertices. We mention that this algorithm is basically identical to the best known (unparameterized) approximation algorithm for Maximum Disjoint Paths [23, 24].

▶ **Theorem 4.** *There is a polynomial-time algorithm for Maximum Vertex-Disjoint Shortest Paths on directed and weighted graphs that achieves an approximation factor of* $\min\{\sqrt{n}, \lceil\sqrt{\ell}\rceil\}$.

Proof. Let OPT be a maximum subset of terminal pairs that can be connected by shortest pairwise vertex-disjoint paths and let j be the index of a terminal pair (s_j, t_j) such that a shortest (s_j, t_j)-path contains a minimum number of arcs. We can compute the index j as well as a shortest s_j-t_j-path with a minimum number of arcs by running a folklore modification of Dijkstra's algorithm from each terminal vertex s_i.[4] Let ℓ_j be the number of arcs in the

[4] The standard Dijkstra's algorithm is modified by assigning to each vertex a pair of labels: the distance

found path. Our algorithm iteratively picks the shortest s_j-t_j-path using ℓ_j arcs, removes all involved vertices from the graph, recomputes the distance between all terminal pairs, removes all terminal pairs whose distance increased, updates the index j, and recomputes ℓ_j. We distinguish whether $\ell_j + 1 \leq \min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$ or not.

While $\ell_j + 1 \leq \min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$, note that we removed at most $\ell_j + 1$ terminals pairs in OPT. Hence, if $\ell_j + 1 \leq \min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$ holds at every stage, then we connected at least $|\text{OPT}|/\min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$ terminal pairs, that is, we found a $\min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$-approximation.

So assume that at some point $\ell_j > \min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$ and let x be the number of terminal pairs that we already connected by disjoint shortest paths. By the argument above, we have removed at most $x \cdot \min(\sqrt{n}, \lceil\sqrt{\ell}\rceil)$ terminal pairs from OPT thus far. We now make a case distinction whether or not $\sqrt{n} \leq \lceil\sqrt{\ell}\rceil$. If $\ell_j + 1 > \lceil\sqrt{\ell}\rceil \geq \sqrt{n}$, then we note that all remaining paths in OPT contain at least $\sqrt{n}$ vertices each and since the paths are vertex-disjoint, there can be at most $\sqrt{n}$ paths left in OPT. Hence, we can infer that $|\text{OPT}| \leq (x+1) \cdot \sqrt{n}$. Consequently, even though we might remove all remaining terminal pairs in OPT by connecting s_j and t_j, this is still a $\sqrt{n}$-approximation (and a $\lceil\sqrt{\ell}\rceil$-approximation as we assumed $\lceil\sqrt{\ell}\rceil \geq \sqrt{n}$).

If $\ell_j + 1 > \sqrt{n} \geq \lceil\sqrt{\ell}\rceil$, then we note that all remaining paths in OPT contain at least $\ell_j > \lceil\sqrt{\ell}\rceil - 1$ edges each. Moreover, since ℓ_j and $\lceil\sqrt{\ell}\rceil$ are integers, each path contains at least $\lceil\sqrt{\ell}\rceil$ edges each. Since all paths in OPT contain by definition at most ℓ edges combined, the number of paths in OPT is at most $\ell/\lceil\sqrt{\ell}\rceil \leq \lceil\sqrt{\ell}\rceil$. Hence, we can infer in that case that $|\text{OPT}| \leq (x+1) \cdot \lceil\sqrt{\ell}\rceil$. Again, even if we remove all remaining terminal pairs in OPT by connecting s_j and t_j, this is still a $\lceil\sqrt{\ell}\rceil$-approximation (and a $\sqrt{n}$-approximation as we assumed $\sqrt{n} \geq \lceil\sqrt{\ell}\rceil$). This concludes the proof. ◀

4 Exact Algorithms

In this section, we show that Maximum Vertex-Disjoint Shortest Paths is fixed-parameter tractable when parameterized by ℓ, but it does not admit a polynomial kernel. The proof for the first result uses the technique of *color coding* of Alon, Yuster, and Zwick [2]. Imagine we are searching for some structure of size k in a graph. The idea of color coding is to color the vertices (or edges) of the input graph with a set of k colors and then only search for colorful solutions, that is, structures in which all vertices have distinct colors. Of course, this might not yield an optimal solution, but by trying enough different random colorings, one can often get a constant error probability in $f(k) \cdot \text{poly}(n)$ time. Using the standard tool of (n,k)-perfect hash families, these types of algorithms can be derandomized without significant overhead in the running time.

▶ **Theorem 5.** *Maximum Vertex-Disjoint Shortest Paths on weighted and directed graphs can be solved in $2^{O(\ell)}$ poly(n) time.*

Proof. Let $(G, w, (s_1,t_1), \ldots, (s_k,t_k), p)$ be an instance of Maximum Vertex-Disjoint Shortest Paths. First, we guess the value of ℓ by starting with $\ell = p$ and increasing the value of ℓ by one whenever we cannot find a solution with at least p shortest paths and at most ℓ edges. We start with $\ell = p$ as any set of p disjoint paths contains at least ℓ edges. Notice that the total number of vertices in a (potential) solution with p paths is at most $\ell + p$. We use the color-coding technique of Alon, Yuster, and Zwick [2]. We color the

from the terminal and the number of arcs in the corresponding path; then the pairs of labels are compared lexicographically.

vertices of G uniformly at random using $p+\ell$ colors (the set of colors is $[\ell+p]$) and observe that the probability that all the vertices in the paths in a solution have distinct colors is at least $\frac{(p+\ell)!}{(p+\ell)^{(p+\ell)}} \geq e^{-(p+\ell)}$. We say that a solution to the considered instance is *colorful* if distinct paths in the solution have no vertices of the same color. Note that we do not require that the vertices within a path in the solution are colored by distinct/equal colors. The crucial observations are that any colorful solution is a solution and the probability of the existence of a colorful solution for a yes-instance of Maximum Vertex-Disjoint Shortest Paths is at least $e^{-(p+\ell)}$ as any solution in which all vertices receive distinct colors is a colorful solution.

We use dynamic programming over subsets of colors to find a colorful solution. More precisely, we find the minimum number of arcs in a collection $\mathcal{C} = \{P_i\}_{i\in S}$ of p pairwise vertex-disjoint paths for some $S \subseteq [k]$ satisfying the conditions: (i) for each $i \in S$, the path P_i is a shortest path from s_i to t_i and (ii) there are no vertices of distinct paths of the same color.

For a subset $X \subseteq [p+\ell]$ of colors and a positive integer $r \leq p$, we denote by $f[X,r]$ the minimum total number of arcs in r shortest paths connecting distinct terminal pairs such that the paths contains only vertices of colors in X and there are no vertices of distinct paths of the same color. We set $f[X,r] = \infty$ if such a collection of r paths does not exist.

To compute f, if $r = 1$, then let $W \subseteq V$ be the subset of vertices colored by the colors in X. We use Dijkstra's algorithm to find the set $I \subseteq [k]$ of all indices $i \in [k]$ such that the lengths of the shortest s_i-t_i-paths in G and $G[W]$ are the same. If $I = \emptyset$, then we set $f[X,1] = \infty$. Assume that this is not the case. Then, we use the variant of Dijkstra's algorithm mentioned in Theorem 4 to find the index $i \in I$ and a shortest s_i-t_i-path P in $G[W]$ with a minimum number of arcs. Finally, we set $f[X,1]$ to be equal to the number of arcs in P.

For $r \geq 2$, we compute $f[X,r]$ for each $X \subseteq [p+\ell]$ using the recurrence relation

$$f[X,r] = \min_{Y \subset X} \{f[X \setminus Y, r-1] + f[Y,1]\}. \tag{2}$$

The correctness of computing the values of $f[X,1]$ follows from the description and the correctness of recurrence (2) follows from the condition that distinct paths should not have vertices of the same color.

We compute the values $f[X,r]$ in order of increasing $r \in [p]$. Since computing $f[Y,1]$ for a given set Y of colors can be done in polynomial time, we can compute all values in overall $3^{p+\ell}\operatorname{poly}(n)$ time. Once all values $f[X,r]$ are computed, we observe that a colorful solution exists if and only if $f[S,p] \leq \ell$.

If there is a colorful solution, then we conclude that $(G,w,(s_1,t_1),\ldots,(s_k,t_k),p)$ is a yes-instance of Maximum Vertex-Disjoint Shortest Paths. Otherwise, we discard the considered coloring and try another random coloring and iterate. If we fail to find a solution after executing $N = \lceil e^{p+\ell} \rceil$ iterations, we obtain that the probability that $(G,w,(s_1,t_1),\ldots,(s_k,t_k),p)$ is a yes-instance is at most $(1-\frac{1}{e^{p+\ell}})^{e^{p+\ell}} \leq e^{-1}$. Thus, we return that $(G,w,(s_1,t_1),\ldots,(s_k,t_k),p)$ is a no-instance with the error probability upper bounded by $e^{-1} < 1$. Since the running time in each iteration is $3^{p+\ell}\operatorname{poly}(n)$ and $p \leq \ell$, the total running time is in $2^{O(\ell)}\operatorname{poly}(n)$. Note that we do the color coding and dynamic programming for each value between p and the actual value ℓ. However, this only adds an additional factor of $\ell \leq n$ which disappears in the $\operatorname{poly}(n)$.

The above algorithm can be derandomized using the results of Naor, Schulman, and Srinivasan [28] by replacing random colorings by prefect hash families. We refer to the textbook by Cygan et al. [12] for details on this common technique. ◀

The FPT result of Theorem 5 immediately raises the question about the existence of a polynomial kernel. To show that a parameterized problem P does presumably not admit a polynomial kernel, one can use the framework of *cross-compositions*. Given an NP-hard problem L, a polynomial equivalence relation R on the instances of L is an equivalence relation such that (i) one can decide for any two instances in polynomial time whether they belong to the same equivalence class, and (ii) for any finite set S of instances, R partitions the set into at most $\max_{I \in S} \operatorname{poly}(|I|)$ equivalence classes. Given an NP-hard problem L, a parameterized problem P, and a polynomial equivalence relation R on the instances of L, an OR-cross-composition of L into P (with respect to R) is an algorithm that takes q instances $I_1, I_2, \ldots, I_q$ of L belonging to the same equivalence class of R and constructs in $\operatorname{poly}(\sum_{i=1}^{q} |I_i|)$ time an instance (I, ρ) of P such that (i) ρ is polynomially upper-bounded by $\max_{i \in [q]} |I_i| + \log(q)$, and (ii) (I, ρ) is a yes-instance of P if and only if at least one of the instances I_i is a yes-instance of L. If a parameterized problem admits an OR-cross-composition, then it does not admit a polynomial kernel unless NP $\subseteq$ coNP/poly [5].

In order to exclude a polynomial kernel, we first show that a special case of Maximum Vertex-Disjoint Shortest Paths remains NP-hard. We call this special case Layered Vertex-Disjoint Shortest Paths and it is the special case of Vertex-Disjoint Shortest Paths where all edges have weight 1 and the input graph is layered, that is, there is a partition of the vertices into (disjoint) sets $V_1, V_2, \ldots, V_\lambda$ such that all edges $\{u, v\}$ are between two consecutive layers, that is $u \in V_i$ and $v \in V_{i+1}$ or $u \in V_{i+1}$ and $v \in V_i$ for some $i \in [\lambda - 1]$. Moreover, each terminal pair (s_i, t_i) satisfies that $s_i \in V_1$, $t_i \in V_\lambda$, and each shortest path between the two terminals is *monotone*, that is, it contains exactly one vertex of each layer. Layered Vertex-Disjoint Shortest Paths is formally defined as follows.

Layered Vertex-Disjoint Shortest Paths

Input:	A λ-layered graph $G = (V, E)$ with a λ-partition $\{V_1, V_2, \ldots, V_\lambda\}$ of the vertex set, terminal pairs $(s_1, t_1), (s_2, t_2), \ldots, (s_k, t_k)$ with $s_i \in V_1$, $t_i \in V_\lambda$, and $\operatorname{dist}(s_i, t_i) = \lambda - 1$ for all $i \in [\lambda]$.
Question:	Is there a collection $\mathcal{C} = \{P_i\}_{i \in [k]}$ of pairwise vertex-disjoint paths such that P_i is an s_i-t_i-path of length $\lambda - 1$ for all $i \in [k]$?

It is quite straight-forward to prove that Layered Vertex-Disjoint Shortest Paths is NP-complete.

▶ **Proposition 6.** *Layered Vertex-Disjoint Shortest Paths is NP-complete.*

Proof. We focus on the NP-hardness as Layered Vertex-Disjoint Shortest Paths is a special case of Vertex-Disjoint Shortest Paths and therefore clearly in NP. We reduce from 3-Sat. The main part of the reduction is a selection gadget. The gadget consists of a set U of $n + 1$ vertices $u_0, u_1, \ldots, u_n$ and between each pair of consecutive vertices u_{i-1}, u_i, there are two paths with m internal vertices each. Let the set of vertices be $V_i = \{v_1^i, v_2^i, \ldots, v_m^i\}$ and $W_i = \{w_1^i, w_2^i, \ldots, w_m^i\}$. The set of edges in the selection gadget is

$$E = \{\{u_{i-1}, v_1^i\}, \{u_{i-1}, w_1^i\}, \{v_m^i, u_i\}, \{w_m^i, u_i\} \mid i \in [n]\} \\ \cup \{\{v_j^i, v_{j+1}^i\}, \{w_j^i, w_{j+1}^i\} \mid i \in [n] \wedge j \in [m-1]\}.$$

The constructed instance will have $m + 1$ terminal pairs and is depicted in Figure 3. We set $s_{m+1} = u_0$ and $t_{m+1} = u_n$ and we will ensure that any shortest s_{m+1}-t_{m+1}-path contains all vertices in U and for each $i \in [n]$ either all vertices in V_i or all vertices in W_i. These

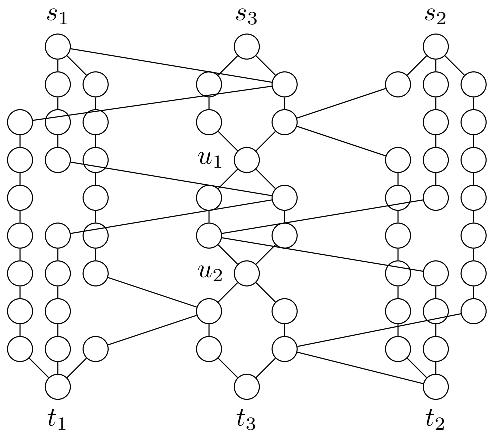

Figure 3 An example of the construction in the proof of Proposition 6 for the input formula $(x_1 \lor x_2 \lor \overline{x_3}) \land (x_1 \lor \overline{x_2} \lor x_3)$.

choices will correspond to setting the i^{th} variable to either true or false. Additionally, we have a terminal pair (s_j, t_j) for each clause C_j. There are (up to) three disjoint paths between s_j and t_j, each of which is of length $n \cdot (m+1)$. These paths correspond to which literal in the clause satisfies it. For each of these paths, let x_i be the variable corresponding to the path. If x_i appears positively in C_j, then we identify the $(i-1)(m+1)+j+1^{\text{st}}$ vertex in the path with w^i_j and if x_i appears negatively, then we identify the vertex with v^i_j. Note that the constructed instance is $(m+1)n$-layered and that once any monotone path starting in s_{m+1} leaves the selection gadget, it cannot end in t_{m+1} as any vertex outside the selection gadget has degree at most two and at the end of these paths are only terminals $t_1, t_2, \ldots, t_m$.

Since the construction runs in polynomial time, we focus on the proof of correctness. If the input formula is satisfiable, then we connect all terminal pairs as follows. Let β be a satisfying assignment. The terminal pair (s_{m+1}, t_{m+1}) is connected by a path containing all vertices in U and for each $i \in [n]$, if β assigns the i^{th} variable to true, then the path contains all vertices in V_i and otherwise all vertices in W_i. For each clause C_j, let x_{i_j} be variable in C_j which β uses to satisfy C_j (if multiple such variables exist, we choose an arbitrary one). By construction, there is a path associated with x_{i_j} that connects s_j and t_j and only uses one vertex in W_i if x_{i_j} appears positively in C_j and a vertex in V_i, otherwise. Since each vertex in V_i and W_i is only associated with at most one such path, we can connect all terminal pairs. For the other direction assume that all $m+1$ terminal pairs can be connected by disjoint shortest paths. As argued above, the s_{m+1}-t_{m+1}-path stays in the selection gadget. We define a truth assignment by assigning the i^{th} variable to true if and only if the s_{m+1}-t_{m+1}-path contains the vertices in V_i. For each clause C_j, we look at the neighbor of s_j in the solution. This vertex belongs to a path of degree-two vertices that at some point joins the selection gadget. By construction, the vertex where this happens is not used by the s_{m+1}-t_{m+1}-path, which guarantees that C_j is satisfied by the corresponding variable. Since all clauses are satisfied by the same assignment, the formula is satisfiable and this concludes the proof. ◀

With the NP-hardness of Layered Vertex-Disjoint Shortest Paths at hand, we can now show that it does not admit a polynomial kernel when parameterized by ℓ by providing an OR-cross-composition from its unparameterized version to the version parameterized by ℓ.

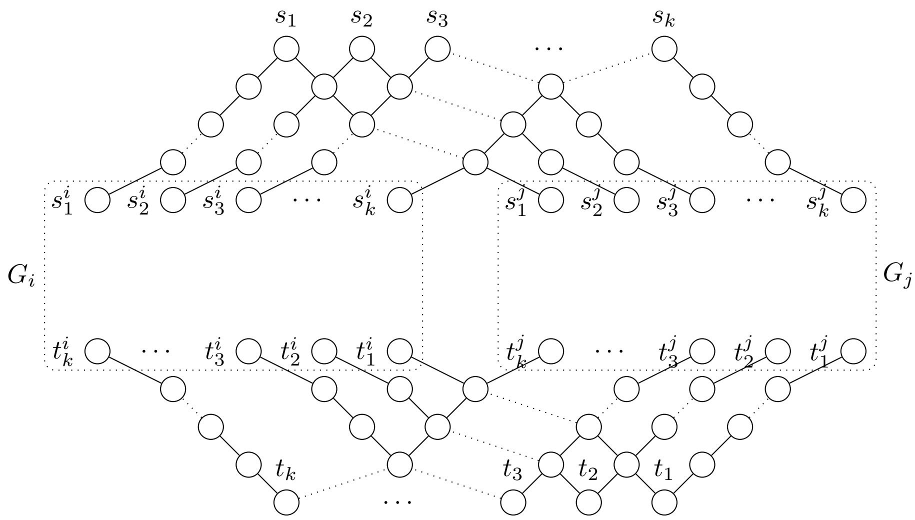

Figure 4 The construction to merge two instances of LAYERED VERTEX-DISJOINT SHORTEST PATHS into one equivalent instance. The dotted edges can be read as regular edges for $k = 4$ and indicate where additional vertices and edges have to be added for more terminal pairs. Note that the height of a vertex in the drawing does not indicate its layer as dotted edges distort the picture.

▶ **Theorem 7.** *LAYERED VERTEX-DISJOINT SHORTEST PATHS parameterized by $\ell = k\cdot(\lambda-1)$ does not admit a polynomial kernel unless NP ⊆ coNP/poly.*

Proof. We present an OR-cross-composition from LAYERED VERTEX-DISJOINT SHORTEST PATHS into LAYERED VERTEX-DISJOINT SHORTEST PATHS parameterized by ℓ. To this end, assume we are given t instances of LAYERED VERTEX-DISJOINT SHORTEST PATHS all of which have the same number λ of layers and the same number k of terminal pairs. Moreover, we assume that t is some power of two. Note that we can pad the instance with at most t trivial no-instances to reach an equivalent instance in which the number of instances is a power of two and the size of all instances combined has at most doubled.

The main ingredient for our proof is a construction to merge two instances into one. The construction is depicted in Figure 4. We first prove that the constructed instance is a yes-instance if and only if at least one of the original instances was a yes-instance. Afterwards, we will show how to use this construction to get an OR-cross-composition for all t instances.

To show that the construction works correctly, first assume that one of the two original instances is a yes-instance. Since both cases are completely symmetrical, assume that there are shortest disjoint paths between all terminal pairs (s_a^i, t_a^i) for all $a \in [k]$ in G_i. Then, we can connect all terminal pairs (s_b, t_b) by using the unique shortest paths between s_b and s_b^i and between t_b^i and t_b for all $b \in [k]$ together with the solution paths inside G_i. Now assume that there is a solution in the constructed instance, that is, there are pairwise vertex-disjoint shortest paths between all terminal pairs (s_b, t_b) for all $b \in [k]$. First assume that the s_1-t_1-path passes through G_i. Then, this path uses the unique shortest path from t_1^i to t_i. Note that this path blocks all paths between t_b^j and vertices in G_j for all $b \neq 1$. Thus, all paths have to pass through the graph G_i. Note that the only possible way to route vertex-disjoint paths from all s-terminals to all s^i terminals and from all t^i-terminals to all t-terminals is to connect s_a to s_a^i and t_a^i to t_a for all $a \in [k]$. This implies that there is a solution that contains vertex-disjoint shortest paths between s_a^i and t_a^i in G_i for all $a \in [k]$, that is, at least one of the two original instances is a yes-instance. The case where the s_1-t_1-path passes through G_j is analogous since the only monotone path from s_1 to a vertex in G_j is the unique shortest s_1-s_1^j-path and this path blocks all monotone paths from s_a to vertices in G_i for all $a \neq 1$.

Note that the constructed graph is layered and that the number of layers is $\lambda + 2k$. Moreover, the size of the new instance is in $O(|G_i| + |G_j| + k^2)$. To complete the reduction, we iteratively half the number of instances by partitioning all instances into arbitrary pairs and merge the two instances in a pair into one instance. After $\log t$ iterations, we are left with a single instance which is a yes-instance if and only if at least one of the t original instances is a yes-instance. The size of the instance is in $O(\sum_{i \in [t]} |G_i| + t \cdot k^2)$ which is clearly polynomial in $\sum_{i \in [t]} |G_i|$ as each instance contains at least k vertices. Moreover, the parameter ℓ in the constructed instance is $k \cdot (\lambda - 1) + 2k \log t$, which is polynomial in $|G_i| + \log t$ for each graph G_i as G_i contains at least one vertex in each of the λ layers and at least k terminal vertices. Thus, all requirements of an OR-cross-composition are met and this concludes the proof. ◀

Note that since Layered Vertex-Disjoint Shortest Paths is a special case of Vertex-Disjoint Shortest Paths (and therefore of Maximum Vertex-Disjoint Shortest Paths), Theorem 7 also excludes polynomial kernels for these problems parameterized by ℓ.

▶ **Corollary 8.** *Vertex-Disjoint Shortest Paths and Maximum Vertex-Disjoint Shortest Paths parameterized by ℓ do not admit polynomial kernels unless NP ⊆ coNP/poly.*

5 Conclusion

In this paper, we studied Maximum Vertex-Disjoint Shortest Paths. We show that there is no $m^{1/2-\varepsilon}$-approximation in polynomial time unless P = NP. Moreover, if FPT ≠ W[1] or assuming the stronger gap-ETH, we show that there are no non-trivial approximations for Maximum Vertex-Disjoint Shortest Paths in $f(k) \cdot \text{poly}(n)$ time. When parameterized by ℓ, there is a simple $\lceil\sqrt{\ell}\rceil$-approximation in polynomial time that matches the $m^{1/2-\varepsilon}$ lower bound as $\ell \leq \min(n, m)$. Finally, we showed that Maximum Vertex-Disjoint Shortest Paths is fixed-parameter tractable when parameterized by ℓ, but it does not admit a polynomial kernel.

A way to combine approximation algorithms and the theory of (polynomial) kernels are *lossy kernels* [26]. Since the exact definition is quite technical and not relevant for this work, we only give an intuitive description. An α-approximate kernel or lossy kernel for an optimization problem is a pair of algorithms that run in polynomial time which are called *pre-processing algorithm* and *solution-lifting algorithm*. The pre-processing algorithm takes as input an instance (I, ρ) of a parameterized problem P and outputs an instance (I', ρ') of P such that $|I'| + \rho' \leq g(\rho)$ for some computable function g. The solution-lifting algorithm takes any solution S of (I', ρ') and transforms it into a solution S^* of (I, ρ) such that if S is a γ-approximation for (I', ρ') for some $\gamma \geq 1$, then S^* is an $\gamma \cdot \alpha$-approximation for (I, ρ). If the size of the kernel is $g(\rho)$ and if g is constant or a polynomial, then we call it a constant-size or polynomial-size α-approximate kernel, respectively. It is known that a (decidable) parameterized problem admits a constant-size approximate α-kernel if and only if the unparameterized problem associated with P can be α-approximated (in polynomial time) [26]. Moreover, any (decidable) parameterized problem admits an α-approximate kernel (of arbitrary size) if and only if the problem can be α-approximated in $f(\rho) \cdot \text{poly}(|I|)$ time.

In terms of lossy kernelization, our results imply that there are no non-trivial lossy kernels for the parameter k. For the parameter ℓ, Theorem 4 implies a constant-size lossy kernel for $\alpha \in \Omega(\sqrt{\ell})$ and Theorem 5 implies an $f(\ell)$-size lossy kernels for any $\alpha \geq 1$. This leaves the following gap which we pose as an open problems.

▶ **Open Problem 1.** *Are there any* $\text{poly}(\ell)$*-size lossy kernels for* Maximum Vertex-Disjoint Shortest Paths *with* $\alpha \in o(\sqrt{\ell})$ *(or even constant* α*)?*

References

1 Shyan Akmal, Virginia Vassilevska Williams, and Nicole Wein. Detecting disjoint shortest paths in linear time and more. In *Proceedings of the 51st International Colloquium on Automata, Languages, and Programming, ICALP*, pages 9:1–9:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPICS.ICALP.2024.9.

2 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *Journal of the ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.

3 Matthias Bentert, André Nichterlein, Malte Renken, and Philipp Zschoche. Using a geometric lens to find k-disjoint shortest paths. *SIAM Journal on Discrete Mathematics*, 37(3):1674–1703, 2023.

4 Kristof Berczi and Yusuke Kobayashi. The directed disjoint shortest paths problem. In *Proceedings of the 25th Annual European Symposium on Algorithms (ESA)*, pages 13:1–13:13. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.ESA.2017.13.

5 Hans L. Bodlaender, Bart M. P. Jansen, and Stefan Kratsch. Kernelization lower bounds by cross-composition. *SIAM Journal on Discrete Mathematics*, 28(1):277–305, 2014. doi:10.1137/120880240.

6 Parinya Chalermsook, Marek Cygan, Guy Kortsarz, Bundit Laekhanukit, Pasin Manurangsi, Danupon Nanongkai, and Luca Trevisan. From gap-exponential time hypothesis to fixed parameter tractable inapproximability: Clique, dominating set, and more. *SIAM Journal on Computing*, 49(4):772–810, 2020. doi:10.1137/18M1166869.

7 Yijia Chen, Yi Feng, Bundit Laekhanukit, and Yanlin Liu. Simple combinatorial construction of the $k^{o(1)}$-lower bound for approximating the parameterized k-clique. *CoRR*, abs/2304.07516, 2023. doi:10.48550/arXiv.2304.07516.

8 Rajesh Chitnis, Samuel Thomas, and Anthony Wirth. Lower bounds for approximate (& exact) k-disjoint-shortest-paths. In *Proceedings of the 22nd International Workshop on Approximation and Online Algorithms (WAOA)*, 2024. Accepted for Publication.

9 Julia Chuzhoy, David H. K. Kim, and Rachit Nimavat. Almost polynomial hardness of node-disjoint paths in grids. *Theory of Computing*, 17:1–57, 2021. doi:10.4086/TOC.2021.V017A006.

10 Julia Chuzhoy, David H. K. Kim, and Rachit Nimavat. New hardness results for routing on disjoint paths. *SIAM Journal on Computing*, 51(2):17–189, 2022. doi:10.1137/17M1146580.

11 Julia Chuzhoy and Shi Li. A polylogarithmic approximation algorithm for edge-disjoint paths with congestion 2. *Journal of the ACM*, 63(5):45:1–45:51, 2016. doi:10.1145/2893472.

12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

13 Irit Dinur. Mildly exponential reduction from gap 3SAT to polynomial-gap label-cover. *Electronic Colloquium on Computational Complexity*, 2016.

14 Tali Eilam-Tzoreff. The disjoint shortest paths problem. *Discrete Applied Mathematics*, 85(2):113–138, 1998. doi:10.1016/S0166-218X(97)00121-2.

15 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization*. Cambridge University Press, Cambridge, 2019.

16 Steven Fortune, John E. Hopcroft, and James Wyllie. The directed subgraph homeomorphism problem. *Theoretical Computer Science*, 10:111–121, 1980. doi:10.1016/0304-3975(80)90009-2.

17 M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

18 Venkatesan Guruswami, Sanjeev Khanna, Rajmohan Rajaraman, F. Bruce Shepherd, and Mihalis Yannakakis. Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems. *Journal of Computer and System Sciences*, 67(3):473–496, 2003. doi:10.1016/S0022-0000(03)00066-7.

19 Johan Håstad. Clique is hard to approximate within $n^{1-\varepsilon}$. In *Proceedings of the 37th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 627–636. IEEE Computer Society, 1996.

20 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-sat. *Journal on Computer and Syststem Sciences*, 62(2):367–375, 2001. doi:10.1006/JCSS.2000.1727.

21 Richard M. Karp. On the computational complexity of combinatorial problems. *Networks*, 5(1):45–68, 1975. doi:10.1002/NET.1975.5.1.45.

22 Karthik C. S. and Subhash Khot. Almost polynomial factor inapproximability for parameterized k-clique. In *Proceedings of the 37th Computational Complexity Conference (CCC)*, pages 6:1–6:21. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICS.CCC.2022.6.

23 Jon M. Kleinberg. *Approximation algorithms for disjoint paths problems*. PhD thesis, Massachusetts Institute of Technology, 1996.

24 Stavros G. Kolliopoulos and Clifford Stein. Approximating disjoint-path problems using packing integer programs. *Mathematical Programming*, 99(1):63–87, 2004. doi:10.1007/S10107-002-0370-6.

25 William Lochet. A polynomial time algorithm for the k-disjoint shortest paths problem. In *Proceedings of the 32nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 169–178. SIAM, 2021. doi:10.1137/1.9781611976465.12.

26 Daniel Lokshtanov, Fahad Panolan, M. S. Ramanujan, and Saket Saurabh. Lossy kernelization. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 224–237. ACM, 2017. doi:10.1145/3055399.3055456.

27 Pasin Manurangsi and Prasad Raghavendra. A birthday repetition theorem and complexity of approximating dense CSPs. In *Proceedings of the 44th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 78:1–78:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.ICALP.2017.78.

28 Moni Naor, Leonard J. Schulman, and Aravind Srinivasan. Splitters and near-optimal derandomization. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 182–191. IEEE Computer Society, 1995. doi:10.1109/SFCS.1995.492475.

29 Neil Robertson and Paul D. Seymour. Graph minors. XIII: The disjoint paths problem. *Journal of Combinatorial Theory. Series B*, 63(1):65–110, 1995. doi:10.1006/JCTB.1995.1006.

30 David Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. *Theory of Computing*, 3(1):103–128, 2007. doi:10.4086/TOC.2007.V003A006.