Proceedings Paper:

Bandopadhyay, S., Banerjee, S., Majumdar, D. et al. (1 more author) (Accepted: 2024) Parameterized Complexity of Shortest Path with Positive Disjunctive Constraints. In: Proceedings of the 17th International Conference on Combinatorial Optimization and Applications (COCOA 2024). 17th International Conference on Combinatorial Optimization and Applications (COCOA 2024), 06-08 Dec 2024, Beijing, China. Lecture Notes in Computer Science . Springer (In Press)

Parameterized Complexity of Shortest Path with Positive Disjunctive Constraints*

Susobhan Bandopadhyay[1][0000−0003−1073−2718], Suman Banerjee[2], Diptapriyo Majumdar[3][0000−0003−2677−4648], and Fahad Panolan[4]

[1] National Institute of Science Education and Research, An OCC of Homi Bhabha National Institute, Bhubaneswar, Odisha, India
susobhan.bandopadhyay@niser.ac.in
[2] Department of Computer Science and Engineering, Indian Institute of Technology, Jammu, India suman.banerjee@iitjammu.ac.in
[3] Indraprastha Institute of Information Technology Delhi, New Delhi, India
diptapriyo@iiitd.ac.in
[4] School of Computer Science, University of Leeds, United Kingdom
F.Panolan@leeds.ac.uk

Abstract. We study the Shortest Path problem subject to positive binary disjunctive constraints. In positive disjunctive constraints, there are certain pairs of edges such that at least one edge from every pair must be part of every feasible solution. We initiate the study of Shortest Path with binary positive disjunctive constrains from the perspective of parameterized complexity. Formally, the input instance is a simple unidrected graph $G = (V, E)$, a forcing graph $G_f = (E, E')$, two vertices $s, t \in V(G)$ and an integer k. Note that the vertex set of G_f is the same as the edge set of G. The goal is to find a set S of at most k edges from G such that there is a path from s to t in the subgraph $G = (V, S)$ and S is a vertex cover in G_f. In this paper, we consider two different natural parameterizations for this problem. One natural parameter is the solution size, i.e. k for which we provide FPT algorithms and polynomial kernelization results. The other natural parameters are structural parameterisations of G_f, i.e. the size of a modulator $X \subseteq E(G) = V(G_f)$ such that $G_f - X$ belongs to some hereditary graph class. We discuss the parameterized complexity of this problem under some structural parameterizations.

Keywords: Shortest Path · Parameterized Complexity · Positive Disjunctive Constraints · Kernelization · Planar Graph.

1 Introduction

In the recent times several classical combinatorial optimization problems on graphs including Maximum Matching, Shortest Path, Steiner Tree have

* Research of Diptapriyo Majumdar has been supported by Science and Engineering Research Board (SERB) grant SRG/2023/001592.

been studied along with some additional binary conjunctive and/or disjunctive constraints [1,7]. Darmann et al. [7] have studied finding shortest paths, minimum spanning trees, and maximum matching of graphs with binary disjunctive constraints in the perspective of classical complexity. These positive or negative binary constraints are defined with respect to pairs of edges. A *negative disjunctive constraint* between an edge-pair e_i and e_j says that both e_i and e_j cannot be present in a feasible solution. A *positive disjunctive constraint* between an edge pair e_i and e_j says that either the edge e_i or the edge e_j or both must be present in any feasible solution. The negative disjunctive constraints can be interpreted as a *conflict graph* such that each vertex of the conflict graph corresponds to an edge in the original graph. Furthermore, for every edge in the conflict graph, at least one endpoint can be part of any feasible solution. Then, a feasible solution must be an independent set in the conflict graph. The positive disjunctive constraints can be interpreted as a *forcing graph* such that each vertex of the forcing graph corresponds to an edge in the original graph. In the forcing graph, each edge must have at least one endpoint included in any feasible solution. Therefore, in the case of positive disjunctive constraints, a feasible solution must be a vertex cover in the forcing graph. Formally, an input to the Forcing-Version of a classical combinatorial optimization problem Π, called as Forcing-Version Π consists of an instance I of Π along with a forcing graph G_f; i.e.; (I, G_f). The vertex set of the forcing graph is the edge set of the original graph. A solution of Forcing-Version Π for the instance (I, G_f) is a solution of I for the original problem along with the property that the solution forms a vertex cover for G_f. To the best of our knowledge, none of the problems SHORTEST PATH, MAXIMUM MATCHING, MINIMUM SPANNING TREE have been explored with positive disjunctive constraints in the perspective of parameterized complexity.

In this paper, we initiate the study of SHORTEST PATH problem with positive disjunctive constraints from the perspective of parameterized complexity and kernelization (see Section 2 for definitions etc). Formally, in the SHORTEST PATH WITH FORCING GRAPH (SPFG) problem, we are given a simple, unweighted graph $G(V, E)$, two vertices s and t, a positive integer k and a forcing graph $G_f(E, E^{'})$. The decision version of this problem asks to check whether there exists a set $E^* \subseteq E(G)$ of at most k edges such that the subgraph induced by the vertex set $V(E^*)$ in G contains an s-t path and also E^* forms a vertex cover in G_f. As "solution size" is the most natural parameter, we formally state the definition of the parameterized version of our problem as follows.

SHORTEST PATH WITH FORCING GRAPH (SPFG) **Parameter:** k
Input: A simple, undirected graph $G(V, E)$, two distinct vertices $s, t \in V(G)$, a positive integer k, and a forcing graph $G_f(E, E^{'})$.
Question: Is there a set E^* of at most k edges from G such that the subgraph $G(V, E^*)$ contains an s-t path in G, and E^* forms a vertex cover in G_f?

In the above definition, the considered parameter is 'solution size'. Darmann et al. [7] have proved that the classical version of Shortest Path with Forcing Graph is NP-hard even when the forcing graph G_f is a graph of degree at most one. So, it can be concluded that even when the forcing graph G_f is very sparse, then also the classical version of Shortest Path with Forcing Graph is NP-hard. In the first part of our paper, we consider Shortest Path with Forcing Graph when parameterized by solution size. Additionally, it is also natural to consider some parameters that are some structures of the input. Observe that the solution size must be as large as the minimum vertex cover size of the forcing graph. If we consider the 'deletion distance of G_f to some hereditary graph class $\mathcal{G}$', then this deletion distance is a parameter that is provably smaller than solution size whenever $\mathcal{G}$ contains a graph that has at least one edge. In the second part of our paper, we also initiate the study of this problem when the considered parameter is deletion distance of G_f to some special graph class. Formally, the definition of this parameterized version is the following.

SPFG-$\mathcal{G}$-Deletion **Parameter:** $|X|$
Input: A simple, undirected graph $G(V, E)$, two distinct vertices $s, t \in V(G)$, a positive integer k, a forcing graph $G_f(E, E^{'})$, a set $X \subseteq E$ such that $G_f - X \in \mathcal{G}$.
Question: Is there a subset E^* of at most k edges from G such that the subgraph $G(V, E^*)$ contains an s-t path in G, and E^* forms a vertex cover in G_f?

Our Contributions: In this paper, we study the Shortest Path with Forcing Graph under the realm of parameterized complexity and kernelization (i.e. polynomial-time preprocessing). We first consider both the original graph G and the forcing graph G_f to be arbitrary graphs and provide FPT and kernelization algorithms. Next, we initiate a systematic study on what happens to the kernelization complexity when either G is a special graph class or G_f is a special graph class. Formally, we provide the following results for Shortest Path with Forcing Graph when the solution size (k) is considered as the parameter.

➢ First, we prove two preliminary results. One preliminary result is a polynomial time algorithm for SPFG when G_f is $2K_2$-free (see Lemma 2). Implication of this result is a dichotomy that SPFG is polynomial-time solvable when G_f is the class of all $2K_2$-free and NP-hard otherwise (see Theorem 1). The other preliminary result is a parameterized algorithm for Shortest Path with Forcing Graph that runs in $\mathcal{O}^*(2^k)$-time[5] (see Theorem 2).

➢ Then, we prove our main result. In particular, we prove that Shortest Path with Forcing Graph admits a kernel with $\mathcal{O}(k^5)$-vertices when G and G_f both are arbitrary graphs (see Theorem 3).

➢ After that, we consider the kernelization complexity of Shortest Path with Forcing Graph when G is a planar graph and G_f is an arbitrary graph. In this condition, we provide a kernel with $\mathcal{O}(k^3)$ vertices (see Theorem 4).

[5] $\mathcal{O}^*$ hides polynomial factor in the input size.

➢ Next, we consider when G is an arbitrary graph and G_f is a graph belonging to a special hereditary graph class. In this paper, we focus on the condition when G_f is a cluster graph (i.e. a disjoint union of cliques) or a bounded degree graph. In both these conditions, we provide a kernel with $\mathcal{O}(k^3)$ vertices for Shortest Path with Forcing Graph (see Theorem 5).

Finally, we consider the SPFG-$\mathcal{G}$-Deletion problem. It follows from the results of Darmann et al. [7] that even if $X = \emptyset$ and G_f is a 2-ladder, i.e. a graph of degree one, SPFG-$\mathcal{G}$-Deletion is NP-hard. Therefore, it is unlikely to expect the possibility that SPFG-$\mathcal{G}$-Deletion would admit an FPT algorithm even when $\mathcal{G}$ is a very sparse graph classes. We complement their NP-hardness result by proving that SPFG-$\mathcal{G}$-Deletion admits an FPT algorithm when $\mathcal{G}$ is the class of all $2K_2$-free graphs (see Theorem 6).

Related Work: Recently, conflict free and forcing variant of several classical combinatorial optimization problem including Maximum Flow [14], Maximum Matching [6], Minimum Spanning Tree [6,7], Set Cover [10], Shortest Path [7] etc have been studied extensively in both algorithmic and complexity theoretic point of view. Recently, some of these problems also have been studied in the realm of parameterized complexity as well [1,12]. Agrawal et al. [1] have studied Shortest Path and Maximum Matching with conflict free version and proved that both the problems are W[1]-hard when parameterized by solution size. They also investigated the complexity of the problems when the conflict graph has some topological structure. Darmann et al. [7] studied both the problems along with both the constraints conflict graph and forcing graph. they showed that the conflict free variant of maximum matching problem is NP-hard even when the conflict graph is a collection of disjoint edges.

Organization of the Paper: We organize our paper as follows. In Section 2, we provide some notations related to graph theory, and parameterized complexity. In the same section, we also prove our first two preliminary results (Theorem 1 and Theorem 2). After that, in Section 3, we prove the main result (Theorem 3) of our paper. Next, in Section 4, we give a short illustration how we can improve the size of our kernels of Section 3 when either the input graph G or G_f belongs to some special graph classes. Additionally, in the same section, we provide a result (Theorem 6) of SPFG on the structural parameterizations for SPFG. Finally, in Section 5, we conclude with open problems and future research directions.

2 Preliminaries

In this section, we describe the notations and symbols used in this paper. Additionally, we also provide some preliminary results for our porblem in this section.

Graph Theory: All the graphs considered in this paper are simple, finite, undirected and unweighted. The notations and terminologies used in this paper are fairly standard and adopted from the Diestel's book of graph theory [8]. In our problem, we are dealing with two different graphs: original graph G and forcing graph G_f. We denote the number of vertices and edges of G by n and m, respectively. Similarly for G_f, it is m and m' (since the edge set of G is same as the vertex set of G_f). Given a graph $G(V,E)$ and an edge $e \in E(G)$, V_e denotes the set of two end vertices of e. For any subset of edges $E^* \subseteq E(G)$, by $V(E^*)$ we denote the set of all the vertices that constitute the edge set; i.e.; $V(E^*) = \bigcup_{e \in E^*} V_e$. Informally, $V(E^*)$ denote the set of all endpoints of the edges in E^*. Given, any two vertices u and v, we denote its shortest path distance by $dist(u,v)$. For any graph $G(V,E)$ a subset of its vertices $S \subseteq V(G)$ is said to be a *vertex cover* of G if every edge of G has at least one of its endpoints in S. A subset of the vertices S is said to be an independent set of G if between any pair of vertices of S, there does not exist any edge in G. For any subset of vertices S of G, the subgraph induced by the vertex set in G is denoted by $G[S]$. Furthermore, for a set of edges F, the graph $G[F]$ is the graph with $G'(V,F)$. Informally, given an edge set $F \subseteq E(G)$, the graph $G[F]$ has vertex set $V(G)$ and the edge set F. For any graph G and any vertex $v \in V(G)$, $G - \{v\}$ denotes the graph that can be obtained by deleting v and the edges incident on it from G. This notion can be extended for a subset of vertices as well. Given a graph $G = (V,E)$ and a set $X \subseteq V(G)$. A graph operation *identification* of the vertex subset X into a new vertex u_X is performed by constructing a graph $\hat{G}$ as follows. First, delete the vertices of X from G and then add a new vertex u_X. Then, for every $v \in N_G(X)$, make vu_X an edge of $\hat{G}$. This graph operation was also defined in Majumdar et al. [13]. A graph is said to be a *cluster graph* if every connected component is a clique. A graph is said to be a *degree-η-graph* if every vertex has degree at most η. A connected graph is said to be $2K_2$-*free* if it does not contain any pair of edges that are nonadjacent to each other. A graph is said to be a *planar graph* if it can be drawn in the surface of a sphere without crossing edges. We use the following property of planar graph in our results.

Proposition 1 ([15]). *If G is a simple planar graph with n vertices, then G has at most $3n-6$ edges.*

A graph is said to be a 2-*ladder* if every connected component is a path of length one. Similarly, a graph is said to be a 3-*ladder* if every connected component is a path of length two.

Parameterized Complexity and Kernelization: A *parameterized problem Π* is denoted as a subset $\Sigma^* \times \mathbb{N}$. An instance to a parameterized problem is denoted by (I,k) where $(I,k) \in \Sigma^* \times \mathbb{N}$ where Σ is a finite set of alphabets and $\mathbb{N}$ is the set of natural numbers. A parameterized problem $\Pi \subseteq \Sigma^* \times \mathbb{N}$ is said to be *fixed-parameter tractable* (or *FPT* in short) if there exists an algorithm $\mathcal{A}$ which runs in $\mathcal{O}(f(k) \cdot |I|^c)$ time where $f(k)$ is a function of k and independent of n and c is a positive constant independent of n and k. We denote the running time $\mathcal{O}(f(k) \cdot |I|^c)$ by the shorthand notation $\mathcal{O}^*(f(k))$ where we suppress the polynomial factors. We adopt the notations and symbols related to parameterized algorithms form the books [9,4]. A parameterized problem Π admits a *kernelization* (or *kernel* in short) if starting with any arbitrary instance (I, k) of the problem, there exists a polynomial-time algorithm that constructs an equivalent instance $(I^{'}, k^{'})$ such that $|I^{'}| + k^{'} \leq g(k)$ for some commutable function $g(\cdot)$. This function $g(\cdot)$ denotes the *size* of the kernel. If $g(k)$ is bounded by a function polynomial in k, then Π is said to admit a *polynomial kernel*. It has been shown by Cai et al. [2] that a problem is in FPT if and only if there exists a kernalization. If a parameterized problem Π admits a kernelization algorithm, we also call that Π admits a *kernel* (in short). We describe the kernalization process by writing a number of reduction rules. A reduction rule takes one instance (say $\mathcal{I}$) of Π and generates the reduced instance (say $\mathcal{I}^{'}$) of Π. We say a reduction rule is *safe* if the following condition holds: "$\mathcal{I}$ is a `Yes`-instance if and only if $\mathcal{I}^{'}$ is a `Yes`-instance." The efficiency of a kernel (or kernelization algorithm) is determined by the size of the kernel. There are many parameterized problems that are fixed-parameter tractable but do not admit polynomial kernels unless NP $\subseteq$ coNP/poly. So, from the perspective of polynomial-time preprocessing, we look for kernels of polynomial-size.

Graph Parameters: In parameterized complexity, though the natural parameterization is the solution size, however, several structural graph parameters have also taken into account [11]. In our problem, the natural parameter is the vertex cover of the forcing graph. As mentioned in [3], the vertex cover can be computed in $\mathcal{O}^*(1.2738^k)$ time where k is the size of the vertex cover. Another important graph parameter is the $\mathcal{G}$-*deletion set* where $\mathcal{G}$ is a graph class. A subset of the vertices $S \subseteq V(G)$ is said to be a *deletion set* to graph class $\mathcal{G}$ if $G - S \in \mathcal{G}$.

Some Preliminary Algorithmic Results: In this section, we establish some classical complexity dichotomy result and some related parameterized complexity results for this problem. The first part of this section gives a proof that the problem is polynomial-time solvable when the forcing graph is a $2K_2$-free graph. Towards this, we define the following annotated problem that would be useful for both the classical and parameterized complexity results.

Ext-SPFG

Input: A simple undirected graph $G(V, E)$, two distinct vertices $s, t \in V(G)$, a forcing graph $G_f(E, E^{'})$ and a vertex cover S of G_f.

Goal: Find a subset $E^* \subseteq E$ with minimum number of edges such that $S \subseteq E^*$, i.e. E^* extends S, and the induced subgraph in G by the edge set E^* contains an s-t path in G.

Our next lemma provides a polynomial-time algorithm for Ext-SPFG.

Lemma 1. *($\star$)* Ext-SPFG *can be solved in polynomial-time.*

[5] Due to lack of space, the proofs that are omitted or marked $\star$ can be found in the full version (`https://arxiv.org/abs/2309.04346`).

Using the above lemma, we can provide a polynomial-time algorithm for SPFG the forcing graph is $2K_2$-free.

Lemma 2. *(⋆) The* Shortest Path with Forcing Graph *can be solved in polynomial time if the forcing graph is $2K_2$-free.*

The above lemma illustrates that if the forcing graph is $2K_2$-free, then the optimization version of the Shortest Path with Forcing Graph can be solved in polynomial-time. Darman et al. [7] proved that Shortest Path with Forcing Graph is NP-Complete even when the forcing graph G_f is a 2-ladder, i.e. graph of degree one. In particular, their construction ensures that there are several $2K_2$s present in the forcing graph as subgraphs. So, we complete this picture by the following dichotomy.

Theorem 1. Shortest Path with Forcing Graph *is polynomial-time solvable when the forcing graph is a $2K_2$-free graph and* NP-Complete *otherwise.*

After discussing the classical complexity of Shortest Path with Forcing Graph, we move on to discuss the parameterized complexity of the same. Since solution size is the most natural parameter; i.e. the number of edges in an optimal solution, we first prove that Shortest Path with Forcing Graph is FPT when parameterized by the solution size. For this purpose, we use the following existing result by Damaschke et al. [5].

Proposition 2. *[5] Given a graph G and positive integer k, all the vertex cover of G of size at most k can be enumerated in $\mathcal{O}(m + 2^k k^2)$ time.*

We prove the following result by using Proposition 2 and Lemma 1.

Theorem 2. *(⋆) The* Shortest Path with Forcing Graph *is fixed-parameter tractable and can be solved in $\mathcal{O}((m + 2^k k^2)(m + n))$ time.*

3 Polynomial Kernels for SPFG

In the previous section, we have discussed that if Shortest Path with Forcing Graph is fixed-parameter tractable when there are no restrictions on the original graph G and the forcing graph G_f, i.e, G and G_f are arbitrary graphs. This section is devoted to the kernalization complexity of Shortest Path with Forcing Graph problem when solution size is considered as the parameter. Our kernelization algorithm has intuitively two parts, "hitting the edges of G_f" and "providing connectivity between s and t in G". As the edges of G are the vertices of G_f, we define the following edge subsets of G.

➢ We put an edge $e \in E(G)$ in H if $deg_{G_f}(e) \geq k + 1$.
➢ We put $e \in E(G)$ in L if $N_{G_f}(e) \subseteq H$.
➢ $R = E(G) \setminus (H \cup L)$.

Notice for any edge $e \in E(G)$, if $N_{G_f}(e) \subseteq H$, then $e \in L$. Hence, we have the following observation.

Observation 1 *If e is an isolated vertex in G_f, then $e \in L$.*

Now, we prove the following lemma that will be one of the important parts in obtaining the kernel.

Lemma 3. *If $I(G, G_f, s, t, k)$ is a* Yes*-instance then $|H| \leq k$ and $G_f[R]$ has at most k^2 edges.*

Proof. By Observation 1, an isolated vertex e of G_f is in L. So, every $e \in R$ has at least one neighbor (with respect to G_f) in R. As any $e \in V(G_f)$ with degree at least $k+1$ in G_f is put in the set H, any $e \in R$ must have at most k neighbors in R. By our hypothesis, $I(G, G_f, s, t, k)$ is a Yes-instance. Hence, there is $E^* \subseteq E(G)$ such that $|E^*| \leq k$ for every $(a, b) \in E(G_f)$, at least one of a and b must be in E^*. If there is $e \in H \setminus E^*$, then at least $k+1$ edges have to be in E^* that is a contradiction to the fact that $I(G, G_f, s, t, k)$ is a Yes-instance. Hence, $H \subseteq E^*$ implying that $|H| \leq k$. Consider the set R. As every $e \in R$ at least one neighbor belongs to R in G_f and at most k neighbors belong to R in G_f. Hence, the number of edges in G_f that are incident to R is at most k^2. Therefore the cardinality of $V(R)$ is at most $2k^2$. ☐

Observe that the edges in H are necessary for any solution of size at most k that certainly "hits every edge of G_f". But, the role of the edges in L are only to provide connectivity between s and t in G. Let E_k be the set of edges in $G_f[H \cup R]$. Recall, $V(E_k) = \{u, v |\ e(u, v) \in E_k\}$. For our convenience, we also add s and t into $V(E_k)$. More formally, $V(E_k) = V(E_k) \cup \{s, t\}$ and let $Y = V(G) \setminus V(E_k)$. We mark some additional vertices from Y using the following marking scheme.

➢ For each pair (x, y) of vertices in $V(E_k)$ compute a shortest x-y path, P_{xy} via the internal vertices of Y in G.

➢ If $P_{x,y}$ has at most k edges, then mark the edges of $P_{x,y}$.

➢ Else $P_{x,y}$ has more than k edges. Then, do not mark any edge.

➢ Finally, for every pair $x, y \in V(E_k)$, mark the edges of a shortest path $Q_{x,y}$ in G when $|Q_{x,y}| \leq k$.

Let $E_t = \bigcup\limits_{x,y \in V(E_k)} (P_{x,y} \cup Q_{x,y})$ be the set of marked edges of G after the completion of the above marking scheme. Consider $E_M = E_t \cup H \cup R$. Formally, E_M be the edges that are in $H \cup R$ as well as in E_t. We denote $G[E_M] = G(V, E_M)$ be the subgraph of G induced by the set of edges in E_M and consider the instance as $I(G[E_M], G_f[E_M], s, t, k)$. Next, we prove the following lemma.

Lemma 4. *The instance $I(G, G_f, s, t, k)$ is a* Yes*-instance if and only if $I(G[E_M], G_f[E_M], s, t, k)$ is a* Yes*-instance.*

Proof. Let us first give the backward direction ($\Leftarrow$) of the proof. First, assume that the instance $I(G[E_M], G_f[E_M], s, t, k)$ is a Yes-instance. One can make a note that edges present in $G[E_M]$ are also in G and in $G_f[E_M]$ as vertices. Suppose that $G[E_M]$ contains a set of edges E^* such that $G[E^*]$ has an s-t path and E^* is a vertex cover in G_f. Then, $H \subseteq E^*$ and hence E^* also forms a vertex cover of G_f. Moreover, an s-t path passing through a (proper) subset of edges in E^* is also an s-t path in G. Hence, $I(G, G_f, s, t, k)$ is a `Yes`-instance.

Next, we focus on proving the forward direction ($\Rightarrow$). Assume that $I(G, G_f, s, t, k)$ is a `Yes`-instance. Let E^* be the solution to the instance $I(G, G_f, s, t, k)$ and let P be an s-t path contained inside the graph induced by $G(V, E^*)$. As $|E^*| \leq k$ is a solution to $I(G, G_f, s, t, k)$, $H \subseteq E^*$. If $E^* \subseteq E_M$, then E^* is a solution to $I(G[E_M], G_f[E_M], s, t, k)$ and we are done. In case some edge $e \in E^* \setminus E_M$ does not belong to any s-t path in $G(V, E^*)$, then clearly such an edge $e \in L$. We just replace that edge e with $\hat{e}$ such that $\hat{e} \in N_{G_f}(e) \cap H$. Consider those edges that belong to some s-t path in $G(V, E^*)$. Consider those subpaths (one at a time) $P^* \subseteq P$ that contains an edge $e \in E^* \setminus E_M$. Observe that P^* has at most k edges and is an x-y path in G for some $x, y \in V(E_k) \cup \{s, t\}$. But, we have marked a shortest path $\hat{P}^*$ from x to y in G (either via the vertices of Y or in G itself). We just replace the edges of P^* by $\hat{P}^*$. As $|\hat{P}^*| \leq |P^*|$, this constructs an s-t walk. Similarly, for other subpaths also, we use the same replacement procedure and eventually construct an s-t walk with at most k edges in $G[E_M]$. As $\hat{E}^*$ provides an s-t in $G[E_M]$, $\hat{E}^*$ is a solution to $I(G[E_M], G_f, s, t, k)$. □

Observe that for every pair of vertices in $V(E_k)$, we have marked a shortest path of length at most k in G. We are ready to prove our final theorem statement.

Theorem 3. *The* Shortest Path with Forcing Graph *admits a kernel with $\mathcal{O}(k^5)$ vertices and edges.*

Proof. Our kernelization algorithm works as follows. First, we compute a partition of $V(G_f) = H \uplus R \uplus L$ as described. From Lemma 3, we have that $H \cup R$ has at most $\mathcal{O}(k^2)$ edges in G_f. After that, we invoke the marking scheme described. Observe that the marking scheme marks a shortest path of length at most k for every pair of vertices $x, y \in V(E_k)$ and put them in E_M. Hence, $|E_M|$ is $\mathcal{O}(k^5)$. From Lemma 4, $I(G, G_f, s, t, k)$ is a `Yes`-instance if and only if $I(G[E_M], G_f[E_M], s, t, k)$ is a `Yes`-instance. Let $W = V(E_M)$, i.e. the vertices that are the endpoints of the edges of E_M in G. We output $(G[W], G_f[E_M], s, t, k)$ as the output instance. As $|E_M|$ is $\mathcal{O}(k^5)$, $|W|$ is also $\mathcal{O}(k^5)$. Therefore, Shortest Path with Forcing Graph admits a kernel with $\mathcal{O}(k^5)$ vertices and edges. □

4 Improved Kernels for Special Graph Classes and Results on Structural Parameters

Consider an input instance $I(G, G_f, s, t, k)$ to Shortest Path with Forcing Graph. This section is devoted to kernelization algorithms when either G or G_f belongs to some special graph class. For both the results, we give a proof sketch here and refer to appendix for more detailed proofs.

Theorem 4. Shortest Path with Forcing Graph *admits a kernel with $\mathcal{O}(k^3)$ vertices when G is a planar graph.*

Proof. (Sketch) The input graph G has a special property satisfying Euler's formula but the forcing graph G_f can be an arbitrary graph. If G is a simple graph with n vertices, then G can have at most $3n-6$ edges. We partition the vertices of G_f, i.e. the edges of $E(G)$ into H, L and R as inn the previous section. Put an edge $e \in E(G)$ into H if $deg_{G_f}(e) > k$. Put $e \in E(G)$ into L if $N_{G_f}(e) \subseteq H$. Define $R = E(G) \setminus (H \cup L)$. Our first step is to invoke Lemma 3, that if $I(G, G_f, s, t, k)$ is a Yes-instance, then $|H| \leq k$ and $G_f[R]$ has at most k^2 edges. Since every vertex of $G_f[H \cup R]$ is incident to some edge, $H \cup R$ has $\mathcal{O}(k^2)$ vertices that are edges of G. Let $V_L \subseteq V(G)$ denote the vertices spanned by the edges of G present in $H \cup R$ and $V_I = V(G) \setminus V_L$. For every pair $\{x, y\}$ of V_L, we define a boolean variable $J_{(\{x,y\})}$ is true if there is a path from x to y in G with internal vertices in V_I and $J_{(\{x,y\})}$ is false otherwise. We prove a structural characterization which says that "there are $3|V_L| - 6$ distinct pairs of vertices $\{x, y\}$ in V_L for which the boolean variable $J_{(\{x,y\})}$ is true". We now consider the a similar marking scheme $\mathsf{MarkPlanar}(G, G_f, s, t, k)$ as before. We give a description of this for the sake of completeness and clarity. For each pair (x, y) of vertices from V_L, compute a shortest path $P_{x,y}$ from x to y that uses only the vertices of V_I as internal vertices. If $P_{x,y}$ has at most k edges, then mark the edges of $P_{x,y}$. Otherwise do not mark any edge of $P_{x,y}$. Finally, for every pair (x, y) from V_L, mark the edges of a shortest path $Q_{x,y}$ from x to y in G if $Q_{x,y}$ has at most k edges. Consider $E_t \subseteq E(G)$, the set of all edges that are marked and let $E_M = E_t \cup H \cup R$. After that, we prove that "the instance $I(G, G_f, s, t, k)$ is equivalent to $I(G[E_M], G_f[E_M], s, t, k)$". Using this, we prove our result (Theorem 4) that SPFG admits a kernel with $\mathcal{O}(k^3)$ vertices when G is a planar graph. □

Theorem 5. Shortest Path with Forcing Graph *admits a kernel with $\mathcal{O}(k^3)$ vertices when G_f is either a cluster graph or a graph with bounded degree.*

Proof. (Sketch) The input graph G is arbitrary here but the forcing graph G_f can be either a cluster graph or a graph of bounded degree. In this situation, we exploit some special properties of cluster graphs or bounded degree graphs as follows. In particular, for the hitting part, we can ensure that $\mathcal{O}(k)$ vertices are sufficient for the hitting part, i.e. to hit all the edges of G_f. An intuition behind this is that if C is a clique in a graph G_f, then at least $|C| - 1$ vertices of C are part of any vertex cover of G_f. We first prove two statements. The first statement says "if G_f is cluster graph and $I(G, G_f, s, t, k)$ is a Yes-instance, then G_f has at most $2k$ vertices that are not isolated in G_f". The second statement says "G_f is a bounded degree graph with maximum degree at most η and $I(G, G_f, s, t, k)$ is a Yes-instance, then G_f has at most $k\eta$ vertices that are not isolated in G_f". Let $V_L^f \subseteq V(G_f)$ be the set of vertices that are not isolated in G_f and they are edges in G and V_L be the set of vertices of G that are the endpoints of these edges of V_L^f. Consider $V_I = V(G) \setminus V_L$. We use a similar marking procedure as before. For each pair (x, y) of vertices from V_L, compute a shortest path $P_{x,y}$ from x to y in G that uses only the vertices of V_I as internal vertices. If $P_{x,y}$ has at most k edges, then mark the edges of $P_{x,y}$. Otherwise, when $P_{x,y}$ has more than k edges, do not mark any edge of $P_{x,y}$. Finally, for each pair (x, y) of vertices from V_L, compute a shortest path $Q_{x,y}$ from x to y in G when $Q_{x,y}$ has at most k edges. Let E_t be the set of all the edges that are marked by the above mentioned marking scheme and let $E_M = E_t \cup V_L^f$. We consider $G[E_M]$ as the output instance and prove that "$I(G, G_f, s, t, k)$ is a Yes-instance if and only if the output instance $I(G[E_M], G_f[E_M], s, t, k)$ is a Yes-instance". Using this, we can prove (Theorem 5) that SPFG admits a kernel with $\mathcal{O}(k^3)$ vertices when G_f is either a cluster graph or a graph of bounded degree. □

Results on Structural Parameterizations Now, we provide a short summary of our result the structural parameterization of Shortest Path with Forcing Graph. We primarily consider the case when the deletion distance (k) to $2K_2$-free graph of G_f. Our first step here is prove the following lemma that enumerates all minimal vertex covers of a $2K_2$-free graph.

Lemma 5. *($\star$) Given an instance (G, G_f, X, s, t, ℓ) to the* SPFG-$2K_2$-Free-Deletion *problem, the set of all minimal vertex covers of G can be enumerated in $2^{|X|}n^{\mathcal{O}(1)}$-time.*

Using the above lemma, we can prove the following theorem saying that SPFG-$2K_2$-Free-Deletion is fixed-parameter tractable.

Theorem 6. *($\star$)* SPFG-$2K_2$-Free-Deletion *admits an algorithm that runs in $2^{|X|}n^{\mathcal{O}(1)}$-time.*

5 Conclusion and Open Problems

In this paper, we have initiated the study of Shortest Path with Forcing Graph under the realm of parameterized complexity. One natural open problem is to see if our kernelization results for Shortest Path with Forcing Graph can be improved, i.e. can we get a kernel with $\mathcal{O}(k^4)$ vertices for SPFG when both G and G_f are arbitrary graphs? We strongly believe that those results can be improved but some other nontrivial techniques might be necessary. In. addition, it would be useful to have a systematic study of this problem under positive disjunctive constraints containing three (or some constant number of) variables. From the perspective of kernelization complexity, we leave the following open problems for future research directions.

➢ Can we get a kernel with $\mathcal{O}(k^4)$ vertices for SPFG when the input graph G is arbitrary graph but the forcing graph G_f is a graph of degeneracy η? Our results only show that if the forcing graph is of bounded degree, then we can get a kernel with $\mathcal{O}(k^3)$ vertices. In fact, even if G_f is a forest, then also it is unclear if we can get a kernel with $\mathcal{O}(k^3)$ or $\mathcal{O}(k^4)$ vertices.

➢ What happens to the kernelization complexity when G_f is an interval graph while G is an arbitrary graph? Can we get a kernel with $\mathcal{O}(k^3)$ vertices in such case?

➢ Finally, can we generalize our result of Theorem 4 when G is a graph of bounded treewidth or graph of bounded degeneracy?

References

1. Agrawal, A., Jain, P., Kanesh, L., Saurabh, S.: Parameterized complexity of conflict-free matchings and paths. Algorithmica pp. 1–27 (2020)
2. Cai, L., Chen, J., Downey, R.G., Fellows, M.R.: Advice classes of parameterized tractability. Annals of pure and applied logic **84**(1), 119–138 (1997)
3. Chen, J., Kanj, I.A., Xia, G.: Improved upper bounds for vertex cover. Theoretical Computer Science **411**(40-42), 3736–3756 (2010)
4. Cygan, M., Fomin, F.V., Kowalik, Ł., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized algorithms, vol. 5. Springer (2015)
5. Damaschke, P.: Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. Theoretical Computer Science **351**(3), 337–350 (2006)
6. Darmann, A., Pferschy, U., Schauer, J.: Determining a minimum spanning tree with disjunctive constraints. In: International Conference on Algorithmic Decision Theory. pp. 414–423. Springer (2009)
7. Darmann, A., Pferschy, U., Schauer, J., Woeginger, G.J.: Paths, trees and matchings under disjunctive constraints. Discrete Applied Mathematics **159**(16), 1726–1735 (2011)
8. Diestel, R.: Graph theory 3rd ed. Graduate texts in mathematics **173** (2005)
9. Downey, R.G., Fellows, M.R.: Parameterized complexity. Springer Science & Business Media (2012)
10. Even, G., Halldórsson, M.M., Kaplan, L., Ron, D.: Scheduling with conflicts: online and offline algorithms. Journal of scheduling **12**(2), 199–224 (2009)
11. Guo, J., Hüffner, F., Niedermeier, R.: A structural view on parameterizing problems: Distance from triviality. In: International Workshop on Parameterized and Exact Computation. pp. 162–173. Springer (2004)
12. Jain, P., Kanesh, L., Misra, P.: Conflict free version of covering problems on graphs: Classical and parameterized. Theory of Computing Systems **64**(6), 1067–1093 (2020)
13. Majumdar, D., Ramanujan, M.S., Saurabh, S.: On the approximate compressibility of connected vertex cover. Algorithmica **82**(10), 2902–2926 (2020)
14. Pferschy, U., Schauer, J.: The maximum flow problem with disjunctive constraints. Journal of Combinatorial Optimization **26**(1), 109–119 (2013)
15. West, D.B.: Introduction to Graph Theory. Pearson Education (2007)