Two-Sets Cut-Uncut on Planar Graphs

Matthias Bentert
University of Bergen, Norway

Pål Grønås Drange
University of Bergen, Norway

Fedor V. Fomin
University of Bergen, Norway

Petr A. Golovach
University of Bergen, Norway

Tuukka Korhonen
University of Bergen, Norway

Abstract

We study Two-Sets Cut-Uncut on planar graphs. Therein, one is given an undirected planar graph G and two disjoint sets S and T of vertices as input. The question is, what is the minimum number of edges to remove from G, such that all vertices in S are separated from all vertices in T, while maintaining that every vertex in S, and respectively in T, stays in the same connected component. We show that this problem can be solved in $2^{|S|+|T|}n^{\mathcal{O}(1)}$ time with a one-sided-error randomized algorithm. Our algorithm implies a polynomial-time algorithm for the network diversion problem on planar graphs, which resolves an open question from the literature. More generally, we show that Two-Sets Cut-Uncut is fixed-parameter tractable when parameterized by the number r of faces in a planar embedding covering the terminals $S \cup T$, by providing a $2^{\mathcal{O}(r)}n^{\mathcal{O}(1)}$-time algorithm.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms; Theory of computation → Graph algorithms analysis; Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases planar graphs, cut-uncut, group-constrained paths

Digital Object Identifier 10.4230/LIPIcs.ICALP.2024.22

Category Track A: Algorithms, Complexity and Games

Related Version *Full Version*: `https://arxiv.org/abs/2305.01314` [2]

Funding The research leading to these results has received funding from the Research Council of Norway via the project BWCA (grant no. 314528) and the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819416).

1 Introduction

A *cut* in a graph $G = (V, E)$ is a partitioning (A, B) of V, and we denote by $\text{cut}_G(A)$ the *cut-set*, that is, the set of edges with one endpoint in A and the other in $B = V \setminus A$. For two disjoint sets of vertices S and T, (A, B) is an *S-T-cut* if $S \subseteq A$ and $T \subseteq B$. We study the following variant of the *cut-uncut problem*.

Two-Sets Cut-Uncut

Input:	A graph G, two disjoint terminal sets $S, T \subseteq V(G)$, and an integer $k \geq 0$.
Task:	Decide whether there exists an S-T-cut (A, B) of G with $\lvert\text{cut}_G(A)\rvert \leq k$ such that the vertices of S are in the same connected component of $G[A]$ and the vertices of T are in the same connected component of $G[B]$.

51st International Colloquium on Automata, Languages, and Programming (ICALP 2024).
Editors: Karl Bringmann, Martin Grohe, Gabriele Puppis, and Ola Svensson;
Article No. 22; pp. 22:1–22:18
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Our interest in Two-Sets Cut-Uncut is two-fold. First, Two-Sets Cut-Uncut is a natural optimization variant of the 2-Disjoint Connected Subgraphs problem that received considerable attention from the graph-algorithms and computational-geometry communities [14, 25, 30, 40, 43, 44]. In this problem, one asks whether, for two given disjoint sets $S, T \subseteq V(G)$, one can find disjoint sets $A_1 \supseteq S$ and $A_2 \supseteq T$ such that the subgraphs of G induced by A_i, $i = 1, 2$, are connected. In Two-Sets Cut-Uncut we not only want to decide whether there are disjoint connected sets containing terminal sets S and T, but also minimize the size of the corresponding cut (if it exists). Van 't Hof et al. [44] showed that 2-Disjoint Connected Subgraphs is NP-complete in general graphs, even if $|S| = 2$, and Gray et al. [25] proved that the problem is NP-complete on planar graphs. This implies that Two-Sets Cut-Uncut is also NP-complete on planar graphs.

Second, Two-Sets Cut-Uncut is closely related to the Network Diversion problem, which has been studied extensively by the operations research and networks communities [10, 11, 19, 22, 29, 36]. In this problem, we are given an undirected graph G, two terminal vertices s and t, an edge $b = uv$, and an integer k. The task is to decide whether it is possible to delete at most k edges such that the edge b will become a bridge with s on one side and t on the other. Equivalently, the task is to decide whether there exists a *minimal* s-t-cut of size at most $k + 1$ containing b. While this problem seems very similar to the classic s-t-Minimum Cut problem, the complexity status of this problem (P vs. NP) is widely open. Let us observe that a polynomial-time algorithm for the special case of Two-Sets Cut-Uncut with $|S| = |T| = 2$ implies a polynomial time algorithm for Network Diversion: There are two cases, either s is in the same component as u, or s is in the same component as v, and these correspond to instances of Two-Sets Cut-Uncut with $S = \{s, u\}$ and $T = \{t, v\}$ and $S = \{s, v\}$ and $T = \{t, u\}$, respectively.

Network Diversion has important applications in transportation networks and has therefore also been studied on planar graphs. Cullenbine et al. [10] gave a polynomial-time algorithm for Network Diversion on planar graphs for the special case when both terminals s and t are located on the same face. They posed as an open problem whether this polynomial-time algorithm can be generalized to work on arbitrary planar graphs [10]. Duan et al. put out a preprint [18], which among other results, claims an algorithm resolving Network Diversion on planar graphs in polynomial time, but without a description of the algorithm. We were not able to verify the correctness of the result due to several missing details. The result, however, is an immediate consequence of our main contribution, Theorem 1, establishing the fixed-parameter tractability of Two-Sets Cut-Uncut on planar graphs parameterized by $|S| + |T|$. Theorem 1 also establishes a more general result about fixed-parameter tractability of the problem parameterized by the minimum number of faces r of the graph containing all terminals. (Notice that r never exceeds $|S| + |T|$.)

▶ **Theorem 1.** *There is a one-sided-error randomized algorithm solving Two-Sets Cut-Uncut on planar graphs in $2^{|S|+|T|} \cdot n^{\mathcal{O}(1)}$ time. Moreover, there is a one-sided-error randomized algorithm solving the problem in $2^{\mathcal{O}(r)} \cdot n^{\mathcal{O}(1)}$ time, where r is the number of faces needed to cover $S \cup T$ in a planar embedding.*

Theorem 1 provides the first polynomial-time algorithm for Two-Sets Cut-Uncut on planar graphs for non-singleton S and T. Duan and Xu [19] showed how to solve Two-Sets Cut-Uncut on planar graphs for $|S| = 1$ and $|T| = 2$. This was later extended by Bezáková and Langley [4], who present an $O(n^4)$-time algorithm for $|S| = 1$ and arbitrary T on planar graphs. However, the polynomial time solvability of the case $|S| = |T| = 2$ (which is a generalization of Network Diversion) remained open.

The main tool we develop for showing Theorem 1 is a new algorithmic result about computing shortest paths in group-labeled graphs. We believe this new result to be of independent interest. The group that we consider is the *Boolean group* $(\mathbb{Z}_2^d, +)$, consisting of length-d binary vectors, where the operation $+$ is the component-wise exclusive or (xor). Our algorithm finds a shortest s-t-path in a graph, whose edges are labeled by elements of $(\mathbb{Z}_2^d, +)$ such that the sum of the labels assigned to the edges of the path equals a given value. Furthermore, we impose the constraint that the path can visit certain sets of vertices only once. Formally, we consider the following problem.

Xor-Constrained Shortest Path

Input: A graph G, two vertices s and t, an edge labeling function $g\colon E(G) \to \mathbb{Z}_2^d$, a value $c \in \mathbb{Z}_2^d$, and p sets of vertices $X_1, \ldots, X_p \subseteq V$.

Task: Find an s-t-path P in G that satisfies

(i) $\sum_{e \in E(P)} g(e) = c$, and

(ii) for each $i \in [p]$, $|V(P) \cap X_i| \leq 1$,

and among all such paths minimizes the length.

In Section 3, we give an algorithm for Xor-Constrained Shortest Path that in fact works for general graphs instead of only planar graphs. The result is the following theorem.

▶ **Theorem 2.** *Xor-Constrained Shortest Path can be solved in* $2^{d+p} \cdot (n+m)^{\mathcal{O}(1)}$ *time by a one-sided-error randomized algorithm.*

We call the problem variant where we replace path by cycle in the above problem definition Xor-Constrained Shortest Cycle. We will later show that Theorem 2 directly implies an algorithm for Xor-Constrained Shortest Cycle with the same running time.

The proof of Theorem 2 is based on enhancing the technique introduced by Björklund, Husfeldt, and Taslaman [7] for the T-cycle problem. In T-cycle, the task is to find a shortest cycle that visits a list of specified vertices $T \subseteq V(G)$[1], and Björklund et al. gave a $2^{|T|}n^{\mathcal{O}(1)}$-time algorithm for it. Our algorithm generalizes the algorithm of Björklund et al., because T-cycle can be reduced to Xor-Constrained Shortest Cycle with $d = |T|$ and $p = 0$ as follows. We assign each vertex $v \in T$ to one dimension of $\mathbb{Z}_2^d$, and to enforce that the cycle passes through v, we add a true twin u of v, that is, a vertex adjacent to v with the same neighborhood as v to the graph and assign the edge uv the vector in $\mathbb{Z}_2^d$ that has 1 at only the dimension assigned to v. All other edges are assigned the zero vector $\mathbf{0}$. A cycle evaluating to the all-one vector corresponds to a cycle that visits all vertices in T.

Related work. Besides the closely related work on Network Diversion, 2-Disjoint Connected Subgraphs, and Two-Sets Cut-Uncut that we already mentioned above, let us briefly go through other relevant work.

Two-Sets Cut-Uncut is a special case of Multiway Cut-Uncut, where for a given equivalence relation on the set of terminals, the task is to find a cut (or node-cut) separating terminals according to the relation. This problem is well-studied in parameterized complexity [8, 13, 41]. However, all previous work on parameterized algorithms for Multiway Cut-Uncut has focused on parameterizing by the size of the cut.

[1] The algorithm can also take a list of edges as input by subdividing each target edge and add the new vertex to T.

Multiway Cut is also one of the closest relatives of our problem. Here for a given set of k terminals, one looks for a minimum number of edges separating *all* terminals. On planar graph, the seminal paper of Dahlhaus et al. [15] provides an algorithm of running time $n^{\mathcal{O}(k)}$. Klein and Marx [32] improve the running time to $n^{\mathcal{O}(\sqrt{k})}$ and Marx [37] shows that this running time is optimal (assuming the Exponential Time Hypothesis (ETH)).

The second part of Theorem 1 concerns the parameterization by the number of faces covering the terminal vertices. Such parameterization comes naturally for optimization problems about connecting or separating terminals in planar graphs. In particular, parameterization by the face cover was investigated for Multiway Cut [39], Steiner Tree [3, 31], and various flow problems [21, 23, 35].

Our Theorem 2 belongs to the intersection of two areas around paths in graphs. The first area is about polynomial-time algorithms computing shortest paths in group-labeled graphs [16, 26, 33]. Recently, Iwata and Yamaguchi [28] gave an algorithm for shortest *non-zero* paths in arbitrary group-labeled paths. However, for our purposes, we need an algorithm computing a shortest path whose labels sum to a *specific* element of the group.

The second area is about FPT algorithms for finding paths in graphs satisfying certain properties [6, 7, 24, 20, 34, 45]. As mentioned above, our algorithm for Xor-Constrained Shortest Path can be seen as an extension of the algorithm of Björklund, Husfeldt, and Taslaman [7] for the T-cycle problem to a group-labeled setting.

Organization. The remainder of the article is organized as follows. We start with a general overview of how we achieve our two main results. We then present some notation and necessary definitions in Section 2. Afterwards, we show how to solve Xor-Constrained Shortest Path in Section 3. In Section 4, we apply this algorithm to show Theorem 1 by developing a (randomized) FPT-time algorithm for Two-Sets Cut-Uncut parameterized by the minimum number of faces such that each terminal vertex is incident to at least one such face. Section 5 is devoted to showing that Two-Sets Cut-Uncut is W[1]-hard on general graphs when parameterized by the number of terminals. We conclude in Section 6 with several open problems. Due to space constraints, proofs of some statements are omitted in this extended abstract. These statements are marked ($\star$). The detailed proofs can be found in the full arXiv version of the paper [2]. In the full version, we also present two applications of our FPT-time algorithm to generalize known results from the literature.

1.1 Outline of the Proofs for Theorems 1 and 2

In this section, we outline the proofs of Theorems 1 and 2. For Theorem 1, we first outline the $2^{|S|+|T|} \cdot n^{\mathcal{O}(1)}$-time algorithm for planar Two-Sets Cut-Uncut and then discuss the setting when $S \cup T$ can be covered by at most r faces. Then we consider Theorem 2.

We observe that any optimal solution to Two-Sets Cut-Uncut is an (inclusion-wise) minimal cut in the graph G. Our algorithm is based on the relation between minimal cuts in a planar graph and cycles in its dual graph (see Figure 1). In particular, a set of edges $C \subseteq E(G)$ is a cut-set of a minimal cut in G if and only if in the dual graph G^*, the corresponding set $C^* \subseteq E(G^*)$ is a (simple) cycle. Now, to translate Two-Sets Cut-Uncut into a problem about finding a cycle C^* in G^*, we wish to understand, based on C^*, whether two terminal vertices u and v are on the same side of the cut C in G or on different sides. For this, we observe that if $P \subseteq E(G)$ is the set of edges of an (arbitrary) u-v-path in G and P^* is the corresponding set of edges in G^*, then u and v are on different sides of C if and only if $|C^* \cap P^*|$ is odd.

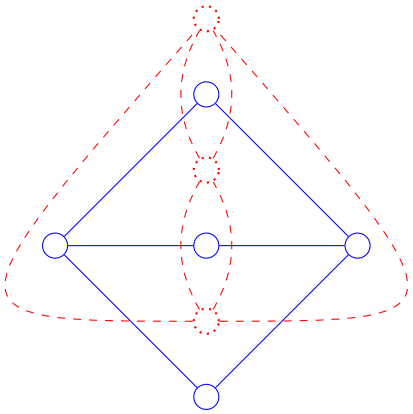

Figure 1 An example of a plane graph (blue) and its dual (multi)graph (dashed/red). Notice that there are bijections between the faces and the vertices, and also between the edges, that is, there is exactly one blue vertex in each red face, one red vertex in each blue face, and each red edge intersects exactly one blue edge and vice versa.

It follows that a constraint stating that u_i and v_i should be on the same/different side of the cut C in G can be expressed as a constraint stating that $|C^* \cap P_i^*|$ should be even/odd for some $P_i^* \subseteq E(G^*)$. By selecting one vertex v in the set of terminals S and writing a "same side" constraint with every other terminal vertex in S and a "different side" constraint with every terminal vertex in T, the Two-Sets Cut-Uncut problem reduces to the problem of finding a shortest cycle C^* in G^* that satisfies $|S| + |T| - 1$ given constraints, each requiring that $|C^* \cap P_i^*| \equiv b_i \pmod 2$ for some $P_i^* \subseteq E(G^*)$ and $b_i \in \{0, 1\}$.

This problem can be equivalently phrased as the Xor-Constrained Shortest Cycle problem with $d = |S| + |T| - 1$, and therefore Theorem 2 indeed implies a $2^{|S|+|T|} \cdot n^{\mathcal{O}(1)}$-time algorithm for Two-Sets Cut-Uncut on planar graphs. Note that here we did not use the condition (ii) in the statement of the Xor-Constrained Shortest Path problem; this condition will be used only for the algorithm utilizing the face cover.

Next, we turn to the more general setting when $S \cup T$ can be covered by at most r faces in a planar embedding of G. First, we observe that by the results of Bienstock and Monma [5], we can decide in $2^{\mathcal{O}(r)} \cdot n$ time whether the input graph has a planar embedding such that the terminals can be covered by at most r faces. Thus, we can assume that G is a plane graph and we are given a face cover of $S \cup T$. Further, we observe that it can be assumed without loss of generality that the input graph is 2-connected. This assumption simplifies arguments because the boundary of each face of a plane 2-connected graph is a cycle [17].

Suppose that f is a face of G that covers some terminals and let C' be the cycle forming the frontier of f. We use the following crucial observation: for the cut-set $C \subseteq E(G)$ of any minimal cut in G separating S and T, it holds that (i) if C' contains vertices of both sets of terminals, then $C \cap E(C')$ separates C' into two connected components (paths) such that each component contains the vertices of exactly one set of terminals, and (ii) if C' contains vertices of one set, then either $C \cap E(C') = \emptyset$ or $C \cap E(C')$ separates C' into two connected components (paths) such that the terminals are in the same component. We use this observation to restrict the behavior of the cycle C^* in G^* corresponding to a potential solution cut-set C. In case (i), we simply delete the edges of G^* that correspond to the edges of C' that should not participate in C (see Figure 2 (a) in the proof of Theorem 1). Case (ii) is more complicated. Suppose that C' contains q terminals. We find q internally vertex disjoint paths $P_1, \ldots, P_q$ in C' whose end-vertices are the terminals. Then we "split"

the vertex f of G^* into q vertices $f_1, \ldots, f_q$ in such a way that each f_i is incident to the edges of G^* corresponding to the edges of P_i (see Figure 2 (b)). However, this splitting would allow a cycle in the dual graph to visit the face f several times. To forbid this, we define $X_f = \{f_1, \ldots, f_q\}$ as used in constraint (ii) of Xor-Constrained Shortest Path and this is the reason why we need constraint (ii) in the problem definition.

We perform the modifications of G^* for all the faces in the cover. This allows us to restrict the number of terminals that we should separate. We pick representatives for each face f in the cover. If the frontier cycle C' of f contains terminals from both sets, we chose one representative from each set from the terminals on C'. If C' contains terminals from one set, we choose one representative. We then apply the same algorithm as for the parameterization by $|S| + |T|$. The difference is that we work only with the representatives and add constraint (ii) to the auxiliary instance of Xor-Constrained Shortest Path given by the sets constructed for the faces from the cover.

We conclude by sketching the main ideas of the algorithm from Theorem 2 for Xor-Constrained Shortest Path. This algorithm works not only on planar graphs but also on general graphs, and it is a generalization of the algorithm by Björklund, Husfeldt, and Taslaman [7] for the T-cycle problem. Our algorithm, like many previous parameterized algorithms for finding paths in graphs [6, 7, 24, 34, 45], exploits the cancellation of monomials in polynomials over fields of characteristic two and randomized polynomial identity testing [42, 46].

The idea of our algorithm is to associate with the input a polynomial over a finite field of characteristic two, and argue that (1) this polynomial is non-zero if and only if a solution exists, and (2) given an assignment of values to variables of the polynomial, the value of the polynomial can be evaluated in $2^{d+p} \cdot n^{\mathcal{O}(1)}$ time.[2] By the DeMillo–Lipton–Schwartz–Zippel lemma [42, 46], the problem can then be solved in $2^{d+p} \cdot n^{\mathcal{O}(1)}$ time by evaluating the polynomial for a random assignment of values. Note that solving the decision version also allows to recover the solution by self-reduction.

In more detail, the polynomial associated with the input is defined as follows. Let us assume that the input graph is a simple graph, as the problem on multigraphs can easily be reduced to simple graphs. For each edge $e \in E(G)$ of the input graph, we associate a variable $f(e)$, and then, for an s-t-walk $W = (e_1, e_2, \ldots, e_\ell)$ of length ℓ, we associate a monomial $f(W) = \prod_{i=1}^{\ell} f(e_i)$. For an integer ℓ, we let $\mathcal{C}_\ell$ denote the set of all s-t-walks of length ℓ that satisfy the conditions (i) and (ii) of the statement of Xor-Constrained Shortest Path, and finally let $f(\mathcal{C}_\ell) = \sum_{W \in \mathcal{C}_\ell} f(W)$ be the polynomial associated with the input. As the monomials of $f(\mathcal{C}_\ell)$ correspond to walks instead of paths, it is not complicated to design a $2^{d+p} \cdot n^{\mathcal{O}(1)}$-time dynamic program for evaluating the value of $f(\mathcal{C}_\ell)$. A more technical part of the proof is to argue that the polynomial $f(\mathcal{C}_\ell)$ is non-zero if and only if a solution exists, in particular, that monomials corresponding to walks that are not paths cancel each other out. This argument is a generalization of the argument used by Björklund et al. [7].

2 Preliminaries

For integers a and b, we use $[a, b]$ to denote the set $\{a, a+1, \ldots, b\}$ and $[b]$ to denote the set $[1, b]$.

[2] Recall that p is the number of constraints for condition (ii) in the Xor-Constrained Shortest Path problem.

Graphs. In this paper, we consider undirected multigraphs, that is, we allow multiple edges and self-loops. We use standard graph-theoretic notation and refer to the textbook by Diestel [17] for undefined notions. Let $G = (V, E)$ be an undirected graph. We use $V(G)$ and $E(G)$ to denote the set of vertices and the set of edges of G, respectively. We use n and m to denote the number of vertices and edges in G, respectively. A *path* P is a graph with vertex set $\{v_0, v_1, \ldots, v_\ell\}$ and edge set $\{v_{i-1}v_i \mid i \in [\ell]\}$. The vertices v_0 and v_ℓ are called the endpoints of P. A *cycle* C is a path with an additional edge between the two endpoints. The length of a path or a cycle is the number of edges in it. For a vertex subset $U \subseteq V$, we use $G[U]$ to denote the subgraph of G induced by the vertices in U and $G - U$ to denote $G[V \setminus V']$. For a set of edges $S \subseteq E$, we write $G - S$ to denote the graph obtained from G by deleting the edges of S.

We are mostly interested in *planar* input graphs. We refer the reader to the textbooks of Diestel [17] and Agnarsson and Greenlaw [1] for rigorous introductions. Informally speaking, a graph is planar if it can be drawn on the plane such that its edges do not cross each other. Such a drawing is called a *planar embedding* of the graph and a planar graph with a planar embedding is called a *plane* graph. We note that checking whether a graph is planar and finding a planar embedding can be done in linear time by the classic algorithm of Hopcroft and Tarjan [27]. The *faces* of a plane graph are the regions bounded by a set of edges and that do not contain any other vertices or edges. The vertices and edges on the boundary of a face form its *frontier*.

Given a plane graph $G = (V, E)$ with faces F, its *dual* graph $G^* = (F, E^*)$ (see Figure 1) is defined as follows. The vertices of G^* are the faces of G and for each $e \in E(G)$, G^* has the *dual* edge e^* whose endpoints are either two faces having e on their frontiers or e^* is a self-loop at f if e is in the frontier of exactly one face f (i.e., e is a bridge of G). Observe that G^* is not necessarily simple even if G is a simple graph as the example in Figure 1 shows. We note that G^* is a planar graph that has a plane embedding where each vertex of G^* corresponding to a face f of G is drawn inside f and each dual edge e^* intersects e only once and e^* does not intersect any other edge of G. Throughout this paper, we assume that G^* has such an embedding.

It is crucial for our results that for a connected plane graph G, each minimal cut in G has a one-to-one correspondence to a cycle in G^*. To be more precise, recall that each cycle on the plane has exactly two faces. Then (A, B) is a minimal cut of a plane graph G if and only if there is a cycle C^* in G^* such that the vertices of A are inside one face of C^* and the vertices of B are inside the other face. Furthermore, C^* is formed by the edges e^* that are dual to the edges $e \in \operatorname{cut}(A)$ and the length of C^* is $|\operatorname{cut}(A)|$.

Let G be a plane graph and let G^* be its dual. We say that a path P (a cycle C) in G *crosses* a cycle C^* of G^* in $e \in E(P)$ ($e \in E(C)$, respectively) if C^* contains the edge $e^* \in E^*$ that is dual to e. The *number of crosses* of P and C^* is the number of edges of P where P and C^* cross. We use the following observation.

▶ **Observation 3.** *Let G be a plane graph, let $s, t \in V$, and let P be an s-t-path. For any cycle C^* of G^*, s and t are in distinct faces of C^* if and only if the number of crosses of P and C^* is odd.*

Lastly, given a subset $U \subseteq V$ of vertices in a plane graph G with faces F, a *face cover* of U is a subset $F' \subseteq F$ of faces such that each vertex in U is on the frontier of a face in F'.

Groups. The group $(\mathbb{Z}_2^d, +)$ consists of the set of all length-d binary strings, and the sum of two strings is defined as the bitwise xor of the strings (or addition without carry over). In this regards, it can be seen as the d-dimensional *bitwise xor vector space* $\mathbb{F}_2^d$. It is easy

to see that this is indeed an (abelian) group: (1) The closure property is trivial, since it by definition contains every length-d binary string. (2) Associativity can be seen by a simple case analysis, i.e., $(a \oplus b) \oplus c = a \oplus (b \oplus c)$ is bitwise 1 if and only if there is an odd number of 1s in the bit's position. (3) The identity element is the all $\mathbf{0}$ vector, i.e. $a \oplus \mathbf{0} = a$. (4) The inverse element of a is a itself, i.e., $a \oplus a = \mathbf{0}$.

3 Shortest Paths under Xor Constraints

In Xor-Constrained Shortest Path, we are given a graph G, two vertices s and t, an edge-labeling function $g\colon E(G) \to \mathbb{Z}_2^d$, a value $c \in \mathbb{Z}_2^d$, and p sets of vertices $X_1, \ldots, X_p \subseteq V(G)$. The problem is to find an s-t-path P that satisfies (i) $\sum_{e \in E(P)} g(e) = c$ and (ii) for each $i \in [p]$, $|V(P) \cap X_i| \leq 1$, and among such paths minimizes the number of edges in P. In this section we prove Theorem 2, which we restate here.

▶ **Theorem 2.** *Xor-Constrained Shortest Path can be solved in $2^{d+p} \cdot (n+m)^{\mathcal{O}(1)}$ time by a one-sided-error randomized algorithm.*

As a corollary, we obtain an algorithm for Xor-Constrained Shortest Cycle by guessing one vertex v in a solution, adding a false twin u of v, that is, a non-neighbor of v with the same neighborhood as v to the input graph such that all edges incident to u are assigned value zero, and then asking for a shortest u-v-path satisfying conditions (i) and (ii).

▶ **Corollary 4.** *Xor-Constrained Shortest Cycle can be solved in $2^{d+p} \cdot (n+m)^{\mathcal{O}(1)}$ time by a one-sided-error randomized algorithm.*

3.1 The Algorithm

In the remainder of this section, we assume that the input graph G is a simple graph. Note that an input n-vertex m-edge multigraph can be turned into an $(n+m)$-vertex $2m$-edge simple graph by first removing self-loops, and then subdividing each edge once, giving the label of the edge to one of the subdivision edges and labeling the other subdivision edge with zero. This exactly doubles the length of the solution. We also assume without loss of generality that $s \neq t$.

Let us next introduce some notation. We say that a sequence $(v_0, v_1, \ldots, v_{\ell-1}, v_\ell)$ of $\ell+1$ vertices is an s-t-walk of length ℓ if $v_0 = s$, $v_\ell = t$, and $v_{i-1}v_i \in E(G)$ for each $i \in [\ell]$. Note that unlike a path, a walk can contain a vertex more than once. We say that an s-t-walk is *feasible* if it satisfies analogies of the contraints (i) and (ii), in particular, if

1. $\sum_{i=1}^{\ell} g(v_{i-1}v_i) = c$, and
2. for each $i \in [p]$, there is at most one $j \in [0, \ell]$ such that $v_j \in X_i$.

For an integer $\ell \geq 1$, let $\mathcal{C}_\ell$ denote the set of all feasible s-t-walks of length (exactly) ℓ. We associate with $\mathcal{C}_\ell$ a polynomial as follows.

Let $q = 2^{\lceil \log_2 n \rceil + 1}$, and recall that $\mathrm{GF}(q)$ is a finite field of characteristic 2 and order q. We define a polynomial over $\mathrm{GF}(q)$ as follows. For each edge $uv \in E(G)$ we associate a variable $f(uv)$. Then, for an s-t-walk $W = (v_0, \ldots, v_\ell)$ of length ℓ, we associate the monomial

$$f(W) = \prod_{i=1}^{\ell} f(v_{i-1}v_i), \tag{1}$$

and for the set $\mathcal{C}_\ell$ of all feasible s-t-walks of length ℓ, we associate the polynomial

$$f(\mathcal{C}_\ell) = \sum_{W \in \mathcal{C}_\ell} f(W).$$

Note that the degree of $f(\mathcal{C}_\ell)$ is ℓ. Our algorithm will be based on the following lemma, which will be proven in Section 3.2.

▶ **Lemma 5.** *The length of a shortest s-t-path satisfying (i) and (ii) is equal to the smallest integer ℓ such that $f(\mathcal{C}_\ell)$ is a non-zero polynomial. If no such ℓ exists, then no such s-t-path exists.*

Given Lemma 5, it remains to design an algorithm for testing if $f(\mathcal{C}_\ell)$ is a non-zero polynomial. For this, we use the DeMillo–Lipton–Schwartz–Zippel lemma.

▶ **Lemma 6** ([42, 46])**.** *Let $f(x_1, \ldots, x_n)$ be a non-zero polynomial of degree d over a field $\mathbb{F}$, and let S be a subset of $\mathbb{F}$. If each x_i is independently assigned a uniformly random value from S, then $f(x_1, \ldots, x_n) = 0$ with probability at most $d/|S|$.*

By Lemma 6 to probabilistically test if $f(\mathcal{C}_\ell)$ is non-zero it suffices to evaluate $f(\mathcal{C}_\ell)$ on a random assignment of values from $\mathrm{GF}(q)$ to the variables $f(uv)$. Because the degree of $f(\mathcal{C}_\ell)$ is $\ell \leq n$ and the order of $\mathrm{GF}(q)$ is $q \geq 2n$, this test is correct with probability at least 0.5 whenever $f(\mathcal{C}_\ell)$ is non-zero. Note that if $f(\mathcal{C}_\ell)$ is the zero polynomial, this test is always correct. Next we show that this evaluation can be done efficiently.

▶ **Lemma 7.** *Given an assignment of values to the variables $f(uv)$ for all $uv \in E(G)$, the value of the polynomial $f(\mathcal{C}_\ell)$ can be evaluated in $\mathcal{O}(2^{d+p}n^2\ell)$ time.*

Proof. We evaluate the polynomial by dynamic programming on walks. For $u \in V(G)$, $l \in [0, \ell]$, $y \in \mathbb{Z}_2^d$, and $T \subseteq [p]$, let $\mathcal{C}(u, l, y, T)$ denote the set of s-u-walks $(s = v_0, v_1, \ldots, v_l = u)$ of length l where

- $\sum_{i=1}^{l} g(v_{i-1}v_i) = y$,
- for each $i \in [p] \setminus T$, it holds that $\{v_0, v_1, \ldots, v_l\} \cap X_i = \emptyset$, and
- for each $i \in T$, there exists exactly one $j \in [0, l]$ such that $v_j \in X_i$.

We denote by $f(\mathcal{C}(u, l, y, T))$ the value $\sum_{W \in \mathcal{C}(u,l,y,T)} f(W)$, where $f(W)$ is defined as in Equation (1), with the empty product interpreted as being equal to 1. Now, we have that $f(\mathcal{C}_\ell) = \sum_{T \subseteq [p]} f(\mathcal{C}(t, \ell, c, T))$. It remains to show that the values $f(\mathcal{C}(u, l, y, T))$ can be computed by dynamic programming.

Let $T_v = \{i \in [p] \mid v \in X_i\}$ for each $v \in V(G)$. Then, the values for $l = 0$ are computed by setting $f(\mathcal{C}(s, 0, 0, T_s)) = 1$ and all other values with $l = 0$ to 0. When $l \geq 1$, the values $f(\mathcal{C}(u, l, y, T))$ are computed by dynamic programming from the values for smaller l as follows.

- If $T_u \subseteq T$, then $f(\mathcal{C}(u, l, y, T)) = \sum_{uw \in E(G)} f(uw) \cdot f(\mathcal{C}(w, l-1, y - g(\{u, w\}), T \setminus T_u))$.
- Otherwise, $f(\mathcal{C}(u, l, y, T)) = 0$.

This clearly computes the values correctly, and runs in overall $\mathcal{O}(2^{d+p}n^2\ell)$ time. ◀

Now, our algorithm works by using Lemma 7 to evaluate $f(\mathcal{C}_\ell)$ for random assignments of values to variables $f(uv)$ for increasing values of $\ell \leq n$, and once it evaluates to non-zero, reports that ℓ is the length of the shortest s-t-path satisfying (i) and (ii). If no such $\ell \leq n$ is found, the algorithm reports that no such s-t-path exists. Note that the correctness of the algorithm depends only on the randomness on the evaluation with the correct ℓ, and therefore the algorithm is correct with probability at least 0.5, and never reports a length shorter than the length of a shortest solution. This probability can be exponentially improved by running the algorithm multiple times. To recover the solution, it suffices to use the algorithm to test which edges can be removed from the graph G until G turns into an s-t-path. Clearly, to both recover the solution and to have an exponentially small error probability it suffices to run the algorithm a polynomial number of times, so this finishes the proof of Theorem 2, modulo the proof of Lemma 5 that will be given in the next subsection.

3.2 Proof of Correctness

This section is devoted to the proof of Lemma 5. We first prove the direction that the existence of a solution of length ℓ implies that $f(\mathcal{C}_\ell)$ is non-zero.

▶ **Lemma 8.** *If an s-t-path of length ℓ satisfying (i) and (ii) exists, then $f(\mathcal{C}_\ell)$ is a non-zero polynomial.*

Proof. Let $W = (s = v_0, v_1, \ldots, v_\ell = t)$ be the sequence of vertices on an s-t-path of length ℓ satisfying conditions (i) and (ii). Note that W is a feasible s-t-walk and $W \in \mathcal{C}_\ell$. Since each vertex occurs in the walk W at most once, we observe that W can be determined uniquely from its set of edges, and W is therefore the only walk in $\mathcal{C}_\ell$ with the monomial $f(W) = \prod_{i=1}^{\ell} f(v_{i-1}v_i)$. Thus, the monomial $f(W)$ occurs in the polynomial $f(\mathcal{C}_\ell)$ with coefficient 1, and therefore $f(\mathcal{C}_\ell)$ is non-zero. ◀

It remains to prove that if no solutions of length at most ℓ exists, then $f(\mathcal{C}_\ell)$ is the zero polynomial. For this, let us state our main lemma, but delay its proof for a bit.

▶ **Lemma 9.** *If no s-t-path of length at most ℓ satisfying conditions (i) and (ii) exists, then there exists a function $\phi : \mathcal{C}_\ell \to \mathcal{C}_\ell$ such that for every $W \in \mathcal{C}_\ell$ it holds that*

1. $\phi(\phi(W)) = W$,
2. $\phi(W) \neq W$, *and*
3. $f(\phi(W)) = f(W)$.

Now, assuming Lemma 9, the proof of Lemma 5 can be finished as follows.

▶ **Lemma 10.** *If no s-t-path of length at most ℓ satisfying conditions (i) and (ii) exists, then $f(\mathcal{C}_\ell)$ is the zero polynomial.*

Proof. Let ϕ be the function given by Lemma 9. By properties 1 and 2, the set $\mathcal{C}_\ell$ can be partitioned into pairs $\{W, \phi(W)\}$. Now, property 3 states that $f(W) = f(\phi(W))$ and since $\mathrm{GF}(q)$ is a field of characteristic 2, it holds that $f(W) + f(\phi(W)) = 0$. Thus, $\sum_{W \in \mathcal{C}_\ell} f(W) = 0$. ◀

Putting Lemmas 8 and 10 together implies Lemma 5. It remains to prove Lemma 9.

Proof of Lemma 9. Assume that no s-t-path of length at most ℓ satisfying (i) and (ii) exists. We will define the function $\phi : \mathcal{C}_\ell \to \mathcal{C}_\ell$ explicitly and show that it satisfies all of the required properties. Let $W = (v_0, v_1, \ldots, v_\ell)$ be an s-t-walk in $\mathcal{C}_\ell$. The idea of the definition of ϕ will be to locate a *subwalk* $(v_i, v_{i+1}, \ldots, v_{j-1}, v_j)$ of W where $0 \leq i < j \leq \ell$, and reverse the subwalk, i.e., map the walk

$$W = (v_0, v_1, \ldots, v_{i-1}, v_i, v_{i+1}, \ldots, v_{j-1}, v_j, v_{j+1}, \ldots, v_\ell)$$

into the walk

$$W\overleftarrow{[i,j]} = (v_0, v_1, \ldots, v_{i-1}, v_j, v_{j-1}, \ldots, v_{i+1}, v_i, v_{j+1}, \ldots, v_\ell).$$

In particular, we will have that $\phi(W) = W\overleftarrow{[i,j]}$ for a carefully chosen pair i, j with $0 \leq i < j \leq \ell$. This pair will be chosen so that $v_i = v_j$, which ensures that $W\overleftarrow{[i,j]} \in \mathcal{C}_\ell$ and $f(W\overleftarrow{[i,j]}) = f(W)$ since the multiset of pairs of adjacent vertices in the walk does not change.

It remains to argue that such a pair i, j can be chosen so that the properties $\phi(\phi(W)) = W$ and $\phi(W) \neq W$ hold. Observe that the property $\phi(W) \neq W$ holds if and only if the subwalk from i to j is not a *palindrome*, i.e., a sequence that is the same when reversed. Now we define a process that outputs a pair i, j so that $0 \leq i < j \leq \ell$, $v_i = v_j$, and the subwalk from i to j is not a palindrome.

The process starts by setting $i = j = 0$. Then, it repeats the following: It first selects i to be the smallest integer $i > j$ so that the vertex v_i occurs in the walk in the indices greater than j more than once. If no such i exists, it outputs FAIL. Then, it sets j to be the largest integer so that $v_i = v_j$, in particular, the index of the last occurrence of v_i in the walk. At this point, it is guaranteed that $0 \leq i < j \leq \ell$ and $v_i = v_j$. Now, if the subwalk from i to j is not a palindrome, it outputs the pair i, j. Otherwise, the process repeats.

Observe that the process always outputs either FAIL or a pair i, j with $0 \leq i < j \leq \ell$ and $v_i = v_j$ such that the subwalk from i to j is not a palindrome. We prove that it actually never outputs FAIL.

▷ Claim 11. The process defined above never outputs FAIL.

Proof of Claim 11. Suppose that the process output FAIL and let $i_1 < j_1 < i_2 < \ldots < j_t$ be the sequence of pairs i, j considered during the process. We define the *contracted walk* W' to be the subsequence of $W = (v_0, \ldots, v_\ell)$ obtained by removing the vertices on the indices in $[i_1 + 1, j_1] \cup [i_2 + 1, j_2] \cup \ldots \cup [i_t + 1, j_t]$ from W. In particular, W' is obtained from W by contracting each palindrome $v_{i_k}, \ldots, v_{j_k}$ considered in the process into a single vertex v_{i_k}.

Now, we claim that W' is a feasible s-t-walk of length at most ℓ, and moreover that no vertex occurs more than once in W'. This is a contradiction, because in that case W' would be in fact an s-t-path of length at most ℓ that satisfies conditions (i) and (ii), but we assumed that no such s-t-path exists. We observe that the contracted walk W' is indeed an s-t-walk, because it was obtained from an s-t-walk by contracting subwalks that each start and end in a same vertex. It also clearly has length at most ℓ, and it satisfies the condition (ii) because the multiset of vertices in W' is a subset of the multiset of vertices in W. For condition (i), we observe that if a subwalk $v_i, \ldots, v_j$ is a palindrome, then $\sum_{k=i+1}^{j} g(v_{k-1}v_k) = 0$ because each pair of adjacent vertices occurs an even number of times as G is a simple graph and we are working in the group $\mathbb{Z}_2^d$. Thus, contracting the palindromes does not change the sum of the edge labels on W, and thus W' satisfies condition (i).

Lastly, we argue that no vertex occurs more than once in W'. For the sake of contradiction, suppose that some vertex occurs more than once in W', which in particular implies that there are indices i',j' with $0 \leq i' < j' \leq \ell$ and $v_{i'} = v_{j'}$ that are not in $[i_1 + 1, j_1] \cup [i_2 + 1, j_2] \cup \ldots \cup [i_t + 1, j_t]$. If $i' < i_1$, then this would contradict the choice of i_1, and if $i' = i_1$, then this would contradict the choice of j_1. Similarly, if $j_k < i' \leq i_{k+1}$ for some $1 \leq k < t$, then this would contradict either the choice of i_{k+1} or j_{k+1}, and if $i' > j_t$, then this would contradict the fact that i_t, j_t was the last pair considered by the algorithm. ◁

Now, the function $\phi : \mathcal{C}_\ell \to \mathcal{C}_\ell$ is defined as $\phi(W) = W\overleftarrow{[i, j]}$, where i, j is the pair output by the process described above. We have already proven that $\phi(W) \neq W$ and $f(\phi(W)) = f(W)$, so it remains to prove that $\phi(\phi(W)) = W$. For this, it remains to observe that the operation $W\overleftarrow{[i, j]}$ does not change how the process for selecting i,j behaves, because it does not change the walk before the index i and it does not change the fact that the last occurrence of v_i is at the index j. ◀

This finishes the proof of Theorem 2.

4 Two-Sets Cut-Uncut Parameterized by the Face Cover Number

In this section, we show that Two-Sets Cut-Uncut is FPT when parameterized by the minimum number of faces in a planar embedding of the input graph covering the terminals. We use the following result by Bienstock and Monma [5] showing that a minimum face cover can be found in FPT time when parameterized by the size of a cover.

▶ **Proposition 12** ([5]). *It can be decided in $2^{\mathcal{O}(r)} \cdot n$ time whether for a set of vertices U of a planar graph G and a positive integer r, there is a planar embedding of G such that at most r faces cover U. Furthermore, if such an embedding exists, it can be found together with the face cover of U in the same time.*

By Proposition 12, we can assume that the input graph is plane, that is, we are given its planar embedding and, furthermore, we are given a face cover of the terminals.

Next, we note that we can consider only minimal cuts.

▶ **Observation 13** ($\star$). *Let (G, S, T, k) be an instance of Two-Sets Cut-Uncut where G is a connected graph. Then, (G, S, T, k) is a yes-instance if and only if there is a minimal cut (A, B) of G with $|\mathrm{cut}(A)| \leq k$ such that $S \subseteq A$ and $T \subseteq B$.*

This observation implies that to solve Two-Sets Cut-Uncut on a plane graph G, we have to find a shortest cycle C^* in the dual graph G^* such that the vertices of S and T are in distinct faces of C^*. First, we prove that we can assume without loss of generality that the input graph is 2-connected. This assumption simplifies arguments because the frontier of each face of a plane 2-connected graph is a cycle [17].

▶ **Lemma 14** ($\star$). *There is a polynomial-time algorithm that, given an instance (G, S, T, k) of Two-Sets Cut-Uncut, solves the problem or outputs an equivalent instance (G', S', T', k) of Two-Sets Cut-Uncut where G' is a 2-connected induced subgraph of G. Furthermore, given a planar embedding of G such that $S \cup T$ can be covered by at most r faces, $S' \cup T'$ can be covered in the induced embedding of G' by at most r faces.*

From now on, we assume that the graph of the considered instances of Two-Sets Cut-Uncut is 2-connected. Hence, the dual graph G^* has no loops. Also, since loops are irrelevant for Two-Sets Cut-Uncut, we assume that the input graph has no loops.

We use the following two separation properties for vertices on the frontier of the same face of a graph.

▶ **Lemma 15.** *Let G be a plane graph, let C be the cycle formed by the frontier of a face f of G, and let X and Y be disjoint nonempty sets of vertices in C. Let C^* be any cycle in G^*. Then, the vertices of X and the vertices of Y are in distinct faces of C^* if and only if $f \in V(C^*)$ and C crosses C^* in two edges e_1 and e_2 such that (i) the vertices of X are in the same connected component of $C - \{e_1, e_2\}$, (ii) the vertices of Y are in the same connected component of $C - \{e_1, e_2\}$, and (iii) the vertices of X and the vertices of Y are in distinct connected components of $C - \{e_1, e_2\}$.*

Proof. Suppose that the vertices of X and the vertices of Y are in distinct faces of C^*. Then, f is a vertex of C^*. Let e_1^* and e_2^* be the edges of C^* incident to f and let e_1 and e_2 be the dual edges of e_1^* and e_2^*, respectively. Note that C contains both e_1 and e_2 and C crosses C^* only in these two edges. We have that $C - \{e_1, e_2\}$ has two connected components P_1 and P_2 that are paths. Since C^* separates X and Y, we have that X is fully contained in P_1 or fully contained in P_2 and Y is fully contained in the respective other path. Thus, conditions (i)–(iii) are fulfilled.

For the opposite direction, assume that C crosses C^* in two edges e_1 and e_2 such that conditions (i)–(iii) are fulfilled. Consider an x-y-path P in C for arbitrary $x \in X$ and $y \in Y$ containing e_1 and excluding e_2 that exists by (i)–(iii). The number of crosses of P and C^* is one. Hence, x and y are in distinct faces of C^* by Observation 3. This concludes the proof. ◀

▶ **Lemma 16.** *Let G be a plane graph, let C be the cycle formed by the frontier of a face f of G, and let X be a nonempty sets of vertices in C. Let C^* be any cycle in G^*. Then, the vertices of X are in the same face of C^* if and only if either $f \notin V(C^*)$ and C does not cross C^* or $f \in V(C^*)$ and C crosses C^* in two edges e_1 and e_2 such that the vertices of X are in the same connected component of $C - \{e_1, e_2\}$.*

Proof. Suppose that the vertices of X are in the same face of C^* and assume that C crosses C^*. Then, f is a vertex of C^* and C^* has two edges e_1^* and e_2^* incident to f. We have that C crosses C^* in the edges e_1 and e_2 that are dual to e_1^* and e_2^*, respectively. Since C is a cycle, $C - \{e_1, e_2\}$ has two connected components P_1 and P_2 that are both paths. We show that either $X \subseteq V(P_1)$ or $X \subseteq V(P_2)$. For the sake of contradiction, assume that there are $x, y \in X$ such that $x \in V(P_1)$ and $y \in V(P_2)$. Then, there is an x-y-path P in C that contains e_1 but excludes e_2. The number of crosses of P and C^* is one and x and y are therefore in distinct faces of C^* by Observation 3; a contradiction. Hence, the vertices of X are in the same connected component of $C - \{e_1, e_2\}$.

For the opposite direction, assume that either C does not cross C^* or C crosses C^* in two edges e_1 and e_2 such that the vertices of X are in the same connected component of $C - \{e_1, e_2\}$. In both cases, for any two vertices $x, y \in X$, there is an x-y-path P that does not cross C^*. Then by Observation 3, the vertices of X are in the same face of C^*. This concludes the proof. ◀

We are now in a position to present the main result of this section.

▶ **Theorem 1.** *There is a one-sided-error randomized algorithm solving* Two-Sets Cut-Uncut *on planar graphs in $2^{|S|+|T|} \cdot n^{\mathcal{O}(1)}$ time. Moreover, there is a one-sided-error randomized algorithm solving the problem in $2^{\mathcal{O}(r)} \cdot n^{\mathcal{O}(1)}$ time, where r is the number of faces needed to cover $S \cup T$ in a planar embedding.*

Proof. We show the claim for the parameterization by the size of a face cover of the terminals and then explain how a simplified version of the algorithm can be used for the parameterization by the number of terminals.

Let (G, S, T, k) be an instance of Two-Sets Cut-Uncut where G is planar. We use Proposition 12 to decide whether there is a planar embedding of G such that the set of terminals $S \cup T$ can be covered by at most r faces. When the algorithm reports that such an embedding does not exist, we stop. Otherwise, we obtain an embedding of G in the plane and a set of faces F' of size at most r covering $S \cup T$. From now on, we assume that G is a plane graph. We remind that G can be assumed to be 2-connected by Lemma 14. We use the embedding of G to construct the dual graph G^* together with its embedding. By Observation 13 and duality, our task is to find a cycle C^* in G^* of length at most k such that S and T are in distinct faces of C^*. We find such a cycle using the algorithm for Xor-Constrained Shortest Cycle from Corollary 4.

We partition F' into two sets where $F_1 \subseteq F'$ is the set of faces having vertices from both S and T on their frontiers and $F_2 \subseteq F'$ consists of the faces $f \in F'$ such that the frontier of f contains either only vertices of S or only vertices of T. We modify G^* by analyzing each face $f \in F'$. The ultimate aim of the modification is to reduce the number of considered terminals.

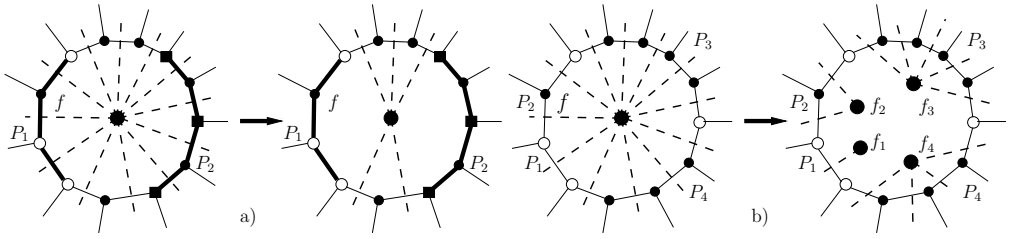

Figure 2 (a) The modification for $f \in F_1$. The vertices of S' are shown by white circles, the vertices of T' are shown by black squares, and the other vertices of G are shown by black circles. The edges of G^* are shown by dashed lines. The paths P_1 and P_2 are shown by thick lines. (b) The modification for $f \in F_2$. The vertices of $L \in \{S, T\}$ are shown by white circles and the other vertices of G are shown by small black circles. The vertex f of G^* and the vertices $f_1, f_2 \ldots, f_4$ are shown by large black circles and the edges of G^* and the constructed new edges are shown by dashed lines.

Modifications for F_1. Let $f \in F_1$ and let C be the cycle of G forming the frontier of f. Recall that $S' = S \cap V(C) \neq \emptyset$ and $T' = T \cap V(C) \neq \emptyset$. If there are no two edges $e_1, e_2 \in E(C)$ such that the vertices of S' and T' are in distinct connected components of $C - \{e_1, e_2\}$, then by Lemma 15, there is no cycle C^* such that the vertices of S' and the vertices of T' are in distinct faces of C^*. This implies that (G, S, T, k) is a no-instance. Hence, we assume that this is not the case and select two inclusion-minimal disjoint paths P_1 and P_2 in C such that $S' \subseteq V(P_1)$ and $T' \subseteq V(P_2)$. We modify G^* by deleting each edge e^* incident to f that is dual to an edge $e \in E(P_1) \cup E(P_2)$ (see Figure 2 (a)).

Modifications for F_2. Let $f \in F_2$, let C be the cycle of G forming the frontier of f, and let $L = V(C) \cap (S \cup T)$. Note that by definition of F_2, either $L \subseteq S$ or $L \subseteq T$. We split the vertex f of G^* into $q = |L|$ vertices $f_1, f_2, \ldots, f_q$ as follows. If $q \geq 2$, then C contains q internally vertex disjoint paths $P_1, P_2, \ldots, P_q$ whose end-vertices are in L (and whose internal vertices are not in L). We then

- delete f and construct a set $X_f = \{f_1, f_2, \ldots, f_q\}$ of q new vertices,
- for each $j \in [q]$ and edge e in P_j, we replace the dual edge e^* of G^* by an edge incident to f_j whose second endpoint is the same as for e^* unless e^* was deleted by some modification for F_1.

For $q = 1$, we set $X_f = \{f\}$ and $f_1 = f$, that is, we do not perform any modification. The construction is shown in Figure 2 (b). Note that the vertices $f_1, f_2, \ldots, f_q$ can be embedded in the face f of G such that the resulting graph H^* is plane. For each edge $e^* \in E(H^*)$, there is an edge $e \in E(G^*)$ such that e^* was constructed from the edge that is dual to e in G^*. Slightly abusing notation, we do not distinguish between the edges of H^* and G^*. In particular, we say that e^* is dual to e.

Next, we assign labels to the edges of H^* from $\mathbb{Z}_2^d$ for some appropriate d. For this, we greedily pick a set R of *representatives* from $S \cup T$ for each $f \in F'$. From each $f \in F_1$, we select two terminals from S and T, respectively, that are on the frontier of the face f of G. For each $f \in F_2$, we pick one terminal from the frontier of the face of f. We then construct an arbitrary inclusion minimal tree Q in G that spans R. This can be done in linear time using standard tools (see, e.g., [9]). We select an arbitrary vertex $u \in R \cap S$ and set $d = |R| - 1$. Observe that $|R| \leq 2|F_1| + |F_2|$ and $d \leq 2|F_1| + |F_2| - 1$. Denote by $v_1, \ldots, v_d$ the vertices of $L \setminus \{u\}$ and let Q_i be the u-v_i-path in Q for each $i \in [d]$. We define $g\colon E(G) \to \mathbb{Z}_2^d$ by setting $g(e) = (\delta_1, \ldots, \delta_d)^\intercal$ where for each $i \in [d]$, $\delta_i = \begin{cases} 1 & \text{if } e \in E(Q_i), \\ 0 & \text{if } e \notin E(Q_i). \end{cases}$

Moreover, let $g^*\colon E(H^*) \to \mathbb{Z}_2^d$ be defined by setting $g^*(e^*) = g(e)$ for each $e^* \in E(H^*)$ that is dual to $e \in E(G)$ and let $c = (c_1, \ldots, c_d)^\intercal \in \mathbb{Z}_2^d$ where

$$c_i = \begin{cases} 0 & \text{if } v_i \in S, \\ 1 & \text{if } v_i \in T \end{cases} \quad \text{for } i \in [d].$$

We show the following claim using Lemma 15 and Lemma 16.

▷ Claim 17 (⋆). The graph G^* contains a cycle C^* of length at most k such that the vertices of S and the vertices of T are in distinct faces of C^* if and only if the instance $(H^*, g^*, c, \{X_f \mid f \in F_2\})$ of Xor-Constrained Shortest Cycle has a solution and the length of a solution cycle is at most k.

By Claim 17, solving Two-Sets Cut-Uncut for (G, S, T, k) is equivalent to solving Xor-Constrained Shortest Cycle for $(H^*, g^*, c, \{X_f \mid f \in F_2\})$. For this, we use the algorithm from Corollary 4.

For the running time, observe that a face cover F' of size r can be constructed in $2^{\mathcal{O}(r)}n$ time by Proposition 12. Given such a cover, the graph H^* together with the sets X_f for $f \in F_2$ can be constructed in polynomial time. Since $d \leq 2|F_1| + |F_2| - 1 \leq 2r$, the labeling g^* and $c \in \mathbb{Z}_2^d$ can also be constructed in polynomial time. Finally, since $d \leq 2|F_1| + |F_2| - 1$ and $p = |\{X_i \mid f \in F_2\}| = |F_2|$, we have that $p + d \leq 2r - 1$ and the algorithm from Corollary 4 runs in $4^r n^{\mathcal{O}(1)}$ time. We conclude that the overall running time is in $2^{\mathcal{O}(r)}n^{\mathcal{O}(1)}$. This concludes the proof.

If we parameterize Two-Sets Cut-Uncut by $\ell = |S| + |T|$, then we can use a simplified variant of the algorithm. Given an instance (G, S, T, k) of Two-Sets Cut-Uncut where G is a planar graph, we use the classic algorithm of Hopcroft and Tarjan [27] to find a plane embedding of G. We then use the variant of the algorithm where we do not modify G^*, that is, we set $H^* = G^*$, and where we assume that all the terminals are representatives, that is, we set $R = S \cup T$. The labeling $g^*\colon E(H^*) \to \mathbb{Z}_2^d$ and c are defined in the same way as in the algorithm for the parameterization by the size of a face cover. By Observation 3, solving Two-Sets Cut-Uncut for (G, S, T, k) is equivalent to solving Xor-Constrained Shortest Cycle for $(H^*, g^*, c, \emptyset)$. Since $d = |R| - 1 = |S| + |T| - 1$, we conclude that we can solve the problem in $2^{|S|+|T|} \cdot n^{\mathcal{O}(1)}$ time using Corollary 4. ◀

5 Hardness

It is known that Two-Sets Cut-Uncut is NP-complete [25] in planar graphs and that it is NP-complete in general graphs even if $|S| = 2$ [44]. We strengthen the latter result by showing that Two-Sets Cut-Uncut remains W[1]-hard parameterized by $|T|$ even if $|S| = 1$ by providing a polynomial-time reduction from Regular Multicolored Clique parameterized by solution size k – a variant of Multicolored Clique where each vertex has the same degree d – such that $|T| = k$. This problem is known to be W[1]-hard and assuming ETH, it cannot be solved in $f(k) \cdot n^{o(k)}$ time [38, 12].

▶ **Proposition 18** (⋆). *Two-Sets Cut-Uncut is* W[1]*-hard when parameterized by* $|T|$ *even if* $|S| = 1$*. Moreover, this restricted version cannot be solved in* $f(|T|) \cdot n^{o(k)}$ *time unless the ETH breaks.*

6 Conclusion

In this paper, we have shown that Two-Sets Cut-Uncut is FPT on planar graphs parameterized by the number of terminals. We have also proven a more general result that the problem remains FPT parameterized by the minimum number of faces required to cover the terminals. Our result implies a polynomial-time algorithm for Network Diversion on planar graphs. We complement this result by showing that Two-Sets Cut-Uncut parameterized by the number of terminals ($|S| + |T|$) is W[1]-hard in general graphs even when $|S| = 1$.

We conclude with a few open problems.

1. First, we repeat the long-standing open question, whether Network Diversion is polynomial-time solvable in general graphs. Even the case for graphs embeddable on a torus remains open.
2. A natural extension of the Two-Sets Cut-Uncut is to extend it to a larger number of sets. Since on general graphs, 3-Way Cut is NP-complete [15], the same holds for Three-Sets Cut-Uncut even when all sets are of size one. However, for planar graphs, k-Way Cut is solvable in polynomial time for fixed k [15, 32, 39]. As a very concrete open question, we ask whether Three-Sets Cut-Uncut is solvable in polynomial time on planar graphs when two sets are of size one and one set is of size two.
3. Our algorithm is randomized and works only on unweighted graphs; can we get rid of either of these restrictions?
4. Finally, is it possible to solve Two-Sets Cut-Uncut in *subexponential* time (in $|S|+|T|$)? In our opinion, this could be a challenging problem. In particular, it is already an open question whether it is possible to find a cycle in a planar graph containing a given set of k terminals in subexponential time (in k).

References

1 Geir Agnarsson and Raymond Greenlaw. *Graph theory: Modeling, applications, and algorithms*. Pearson, 2006.

2 Matthias Bentert, Pål Grønås Drange, Fedor V. Fomin, Petr A. Golovach, and Tuukka Korhonen. Two-sets cut-uncut on planar graphs, 2023. `arXiv:2305.01314`.

3 Marshall Bern. Faster exact algorithms for steiner trees in planar networks. *Networks*, 20(1):109–120, 1990.

4 Ivona Bezáková and Zachary Langley. Minimum planar multi-sink cuts with connectivity priors. In *Proceedings of the 39th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, pages 94–105. Springer, 2014.

5 Daniel Bienstock and Clyde L. Monma. On the complexity of covering vertices by faces in a planar graph. *SIAM Journal on Computing*, 17(1):53–76, 1988.

6 Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. Narrow sieves for parameterized paths and packings. *Journal of Computer and System Sciences*, 87:119–139, 2017.

7 Andreas Björklund, Thore Husfeldt, and Nina Taslaman. Shortest cycle through specified elements. In *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1747–1753. Society for Industrial and Applied Mathematics, 2012.

8 Rajesh Chitnis, Marek Cygan, MohammadTaghi Hajiaghayi, Marcin Pilipczuk, and Michal Pilipczuk. Designing FPT algorithms for cut problems using randomized contractions. *SIAM Journal on Computing*, 45(4):1171–1229, 2016.

9 Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms*. MIT Press, 2009.

10 Christopher A. Cullenbine, R. Kevin Wood, and Alexandra M. Newman. Theoretical and computational advances for network diversion. *Networks*, 62(3):225–242, 2013.

11 Norman D. Curet. The network diversion problem. *Military Operations Research*, 6(2):35–44, 2001.

12 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.

13 Marek Cygan, Pawel Komosa, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, Saket Saurabh, and Magnus Wahlström. Randomized contractions meet lean decompositions. *ACM Transactions on Algorithms*, 17(1):6:1–6:30, 2021.

14 Marek Cygan, Marcin Pilipczuk, Michal Pilipczuk, and Jakub Onufry Wojtaszczyk. Solving the 2-disjoint connected subgraphs problem faster than 2^n. *Algorithmica*, 70(2):195–207, 2014.

15 Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

16 Ulrich Derigs. An efficient Dijkstra-like labeling method for computing shortest odd/even paths. *Information Processing Letters*, 21(5):253–258, 1985.

17 Reinhard Diestel. *Graph Theory*. Springer, 2012.

18 Qi Duan, Haadi Jafarian, Ehab Al-Shaer, and Jinhui Xu. On DDoS attack related minimum cut problems. *CoRR*, abs/1412.3359, 2015. `arXiv:1412.3359`.

19 Qi Duan and Jinhui Xu. On the connectivity preserving minimum cut problem. *Journal of Computer and System Sciences*, 80(4):837–848, 2014.

20 Eduard Eiben, Tomohiro Koana, and Magnus Wahlström. Determinantal sieving. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 377–423. Society for Industrial and Applied Mathematics, 2024.

21 Ranel E. Erickson, Clyde L. Monma, and Arthur F. Jr. Veinott. Send-and-split method for minimum-concave-cost network flows. *Mathematics of Operations Research*, 12(4):634–664, 1987.

22 Ozgur Erken. A branch-and-bound algorithm for the network diversion problem. Master's thesis, Naval Postgraduate School, 2002.

23 Arnold Filtser. A face cover perspective to ℓ_1 embeddings of planar graphs. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1945–1954. Society for Industrial and Applied Mathematics, 2020.

24 Fedor V. Fomin, Petr A. Golovach, Tuukka Korhonen, Kirill Simonov, and Giannos Stamoulis. Fixed-parameter tractability of maximum colored path and beyond. In *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 3700–3712. SIAM, 2023.

25 Chris Gray, Frank Kammer, Maarten Löffler, and Rodrigo I. Silveira. Removing local extrema from imprecise terrains. *Computational Geometry*, 45(7):334–349, 2012.

26 Martin Grötschel and William R. Pulleyblank. Weakly bipartite graphs and the max-cut problem. *Operations Research Letters*, 1(1):23–27, 1981.

27 John E. Hopcroft and Robert E. Tarjan. Efficient planarity testing. *Journal of the ACM*, 21(4):549–568, 1974.

28 Yoichi Iwata and Yutaro Yamaguchi. Finding a shortest non-zero path in group-labeled graphs. *Combinatorica*, 42(S2):1253–1282, 2022.

29 Benjamin S. Kallemyn. *Modeling Network Interdiction Tasks*. Doctoral thesis, Air Force Institute Of Technology, 2015.

30 Frank Kammer and Torsten Tholey. The complexity of minimum convex coloring. *Discrete Applied Mathematics*, 160(6):810–833, 2012.

31 Sándor Kisfaludi-Bak, Jesper Nederlof, and Erik Jan van Leeuwen. Nearly ETH-tight algorithms for planar Steiner tree with terminals on few faces. *ACM Transactions on Algorithms*, 16(3):1–30, 2020.

32 Philip N. Klein and Dániel Marx. Solving planar k-terminal cut in $O(n^{c\sqrt{k}})$ time. In *Proceedings of the 39th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 569–580. Springer, 2012.

33 Yusuke Kobayashi and Sho Toyooka. Finding a shortest non-zero path in group-labeled graphs via permanent computation. *Algorithmica*, 77(4):1128–1142, 2017.

34 Ioannis Koutis. Faster algebraic algorithms for path and packing problems. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 575–586. Springer, 2008.

35 Robert Krauthgamer, James R. Lee, and Havana I. Rika. Flow-cut gaps and face covers in planar graphs. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 525–534. SIAM, 2019.

36 Chungmok Lee, Donghyun Cho, and Sungsoo Park. A combinatorial benders decomposition algorithm for the directed multiflow network diversion problem. *Military Operations Research*, 24(1):23–40, 2019.

37 Dániel Marx. A tight lower bound for planar multiway cut with fixed number of terminals. In *Proceedings of the 39th International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 677–688. Springer, 2012.

38 Luke Mathieson and Stefan Szeider. Editing graphs to satisfy degree constraints: A parameterized approach. *Journal of Computer and System Sciences*, 78(1):179–191, 2012.

39 Sukanya Pandey and Erik Jan van Leeuwen. Planar multiway cut with terminals on few faces. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2032–2063. Society for Industrial and Applied Mathematics, 2022.

40 Daniël Paulusma and Johan M. M. van Rooij. On partitioning a graph into two connected subgraphs. *Theor. Comput. Sci.*, 412(48):6761–6769, 2011. doi:10.1016/j.tcs.2011.09.001.

41 Ashutosh Rai, M. S. Ramanujan, and Saket Saurabh. A parameterized algorithm for mixed-cut. In *Proceedings of the 12th Latin American Symposium (LATIN)*, pages 672–685. Springer, 2016.

42 Jacob T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

43 Jan Arne Telle and Yngve Villanger. Connecting terminals and 2-disjoint connected subgraphs. In *Proceedings of the 39th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 418–428. Springer, 2013.

44 Pim van't Hof, Daniël Paulusma, and Gerhard J. Woeginger. Partitioning graphs into connected parts. *Theoretical Computer Science*, 410(47-49):4834–4843, 2009.

45 Ryan Williams. Finding paths of length k in $O^*(2^k)$ time. *Information Processing Letters*, 109(6):315–318, 2009.

46 Richard Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings of the International Symposiumon Symbolic and Algebraic Computation (EUROSAM)*, pages 216–226. Springer, 1979.