Parameterized Algorithms for Maximum Cut with Connectivity Constraints

Hiroshi Eto
Kyushu University, Fukuoka, Japan
h-eto@econ.kyushu-u.ac.jp

Tesshu Hanaka
Chuo University, Tokyo, Japan
hanaka.91t@g.chuo-u.ac.jp

Yasuaki Kobayashi
Kyoto University, Kyoto, Japan
kobayashi@iip.ist.i.kyoto-u.ac.jp

Yusuke Kobayashi
Kyoto University, Kyoto, Japan
yusuke@kurims.kyoto-u.ac.jp

Abstract

We study two variants of Maximum Cut, which we call Connected Maximum Cut and Maximum Minimal Cut, in this paper. In these problems, given an unweighted graph, the goal is to compute a maximum cut satisfying some connectivity requirements. Both problems are known to be NP-complete even on planar graphs whereas Maximum Cut on planar graphs is solvable in polynomial time. We first show that these problems are NP-complete even on planar bipartite graphs and split graphs. Then we give parameterized algorithms using graph parameters such as clique-width, tree-width, and twin-cover number. Finally, we obtain FPT algorithms with respect to the solution size.

2012 ACM Subject Classification Mathematics of computing → Graph algorithms

Keywords and phrases Maximum cut, Parameterized algorithm, NP-hardness, Graph parameter

Digital Object Identifier 10.4230/LIPIcs.IPEC.2019.13

Related Version A full version of the paper is available at `https://arxiv.org/abs/1908.03389`.

Acknowledgements We thank Akitoshi Kawamura and Yukiko Yamauchi for giving an opportunity to discuss in the Open Problem Seminar at Kyushu University. This work is partially supported by JST CREST JPMJCR1401 and JSPS KAKENHI Grant Numbers JP17H01788, JP18H06469, JP16K16010, JP17K19960, and JP18H05291.

1 Introduction

Maximum Cut is one of the most fundamental problems in theoretical computer science. Given a graph and an integer k, the problem asks for a subset of vertices such that the number of edges having exactly one endpoint in the subset is at least k. This problem was shown to be NP-hard in Karp's seminal work [34]. To overcome this intractability, a lot of researches have been done from various view points, such as approximation algorithms [25], fixed-parameter tractability [40], and special graph classes [7, 9, 20, 28, 29, 37].

In this paper, we study two variants of Maximum Cut, called Connected Maximum Cut and Maximum Minimal Cut. A cut $(S, V \setminus S)$ is *connected* if the subgraph of G induced by S is connected. Given a graph $G = (V, E)$ and an integer k, Connected Maximum Cut is the problem to determine whether there is a connected cut $(S, V \setminus S)$ of size at least k . This problem is defined in [30] and known to be NP-complete even on planar graphs [31] whereas Maximum Cut on planar graphs is solvable in polynomial time [29, 37].

Table 1 The summary of the computational complexity of Maximum Cut and its variants. MC, CMC, and MMC stand for Maximum Cut, Connected Maximum Cut, and Maximum Minimal Cut.

	Graph Class			Parameter				kernel
	Split	Bipartite	Planar	cw	tw	tc	k	k
MC	NP-c [7]	P [trivial]	P [29, 37]	$n^{O(\mathrm{cw})}$ [22]	2^{tw} [7]	2^{tc} [23]	1.2418^k [40]	$O(k)$ [30, 36]
CMC	NP-c [Th. 5]	NP-c [Th. 3]	NP-c [31]	$n^{O(\mathrm{cw})}$ [Th. 17]	3^{tw} [Th. 12]	$2^{2^{\mathrm{tc}}+\mathrm{tc}}$ [Th. 18]	9^k [Th. 22]	No [Th. 24]
MMC	NP-c [Th. 6]	NP-c [Th. 4]	NP-c [30]	$n^{O(\mathrm{cw})}$ [Th. 17]	4^{tw} [Th. 11]	$2^{\mathrm{tc}}3^{2^{\mathrm{tc}}}$ [Th. 19]	$2^{O(k^2)}$ [Th. 21]	No [Th. 24]

Suppose G is connected. We say that a cut $(S, V \setminus S)$ of G is *minimal* if there is no another cut of G whose cutset properly contains the cutset of $(S, V \setminus S)$, where the cutset of a cut is the set of edges between different parts. We can also define minimal cuts for disconnected graphs (See Section 2). Maximum Minimal Cut is the following problem: given a graph $G = (V, E)$ and an integer k, determine the existence of a minimal cut $(S, V \setminus S)$ of size at least k. This type of problems, finding a maximum minimal (or minimum maximal) solution on graphs such as Maximum Minimal Vertex Cover [8, 46], Maximum Minimal Dominating Set [1], Maximum Minimal Edge Cover [35], Maximum Minimal Separator [32], Minimum Maximal matching [24, 45], and Minimum Maximal Independent Set [18], has been long studied.

As a well-known fact, a cut $(S, V \setminus S)$ is minimal if and only if both subgraphs induced by S and $V \setminus S$ are connected when the graph is connected [19]. Therefore, a minimal cut is regarded as a *two-sided* connected cut, while a connected cut is a *one-side* connected cut[1]. Haglin and Venkatean [30] showed that deciding if the input graph has a two-sided connected cut (i.e., a minimal cut) of size at least k is NP-complete even on triconnected cubic planar graphs. This was shown by the fact that for any two-sided connected cut on a connected planar graph G, the cutset corresponds to a cycle on the dual graph of G and vise versa. Hence, the problem is equivalent to the longest cycle problem on planar graphs [30]. Recently, Chaourar proved that Maximum Minimal Cut can be solved in polynomial time on series parallel graphs and graphs without $K_5 \setminus e$ as a minor in [12, 13].

Even though there are many important applications of Connected Maximum Cut and Maximum Minimal Cut such as image segmentation [44], forest planning [11], and computing a market splitting for electricity markets [26], the known results are much fewer than those for Maximum Cut due to the difficult nature of simultaneously maximizing its size and handling the connectivity of a cut.

1.1 Our contribution

Our contribution is summarized in Table 1. We prove that both Connected Maximum Cut and Maximum Minimal Cut are NP-complete even on planar bipartite graphs. Interestingly, although Maximum Cut can be solved in polynomial time on planar graphs [29, 37] and

[1] In [12, 13, 30], the authors used the term "connected cut" for two-sided connected cut. In this paper, however, we use "minimal cut" for two-sided connected cut and "connected cut" for one-sided connected cut for distinction.

bipartite graphs, both problems are intractable even on the intersection of these tractable classes. We also show that the problems are NP-complete on split graphs.

To tackle to this difficulty, we study both problems from the perspective of the parameterized complexity. We give $O^*(\mathrm{tw}^{O(\mathrm{tw})})$-time algorithms for both problems[2], where tw is the tree-width of the input graph. Moreover, we can improve the running time using the rank-based approach [3] to $O^*(c^{\mathrm{tw}})$ for some constant c and using the Cut & Count technique [17] to $O^*(3^{\mathrm{tw}})$ for Connected Maximum Cut and $O^*(4^{\mathrm{tw}})$ for Maximum Minimal cut with randomization. Let us note that our result generalizes the polynomial time algorithms for Maximum Minimal Cut on series parallel graphs and graphs without $K_5 \setminus e$ as a minor due to Chaourar [12, 13] since such graphs are tree-width bounded [42].

Based on these algorithms, we give $O^*(2^{k^{O(1)}})$-time algorithms for both problems. For Connected Maximum Cut, we also give a randomized $O^*(9^k)$-time algorithm. As for polynomial kernelization, we can observe that Connected Maximum Cut and Maximum Minimal Cut admit no polynomial kernel when parameterized by solution size k under a reasonable complexity assumption (see, Theorem 24).

We also consider different structural graph parameters. We design XP-algorithms for both problems when parameterized by clique-width cw. Also, we give $O^*(2^{2^{\mathrm{tc}}+\mathrm{tc}})$-time and $O^*(2^{\mathrm{tc}}3^{2^{\mathrm{tc}}})$-time FPT algorithms for Connected Maximum Cut and Maximum Minimal Cut, respectively, where tc is the minimum size of a twin-cover of the input graph.

1.2 Related work

Maximum Cut is a classical graph optimization problem and there are many applications in practice. The problem is known to be NP-complete even on split graphs, tripartite graphs, co-bipartite graphs, undirected path graphs [7], unit disc graphs [20], and total graphs [28]. On the other hand, it is solvable in polynomial time on bipartite graphs, planar graphs [29, 37], line graphs [28], and proper interval graphs [9]. For the optimization version of Maximum Cut, there is a 0.878-approximation algorithm using semidefinite programming [25]. As for parameterized complexity, Maximum Cut is FPT [40] and has a linear kernel [30, 36] when parameterized by the solution size k. Moreover, the problem is FPT when parameterized by tree-width [7] and twin-cover number [23]. Fomin et al. [22] proved that Maximum Cut is W[1]-hard but XP when parameterized by clique-width.

Connected Maximum Cut was proposed in [30]. The problem is a connected variant of Maximum Cut as with Connected Dominating Set [27] and Connected Vertex Cover [15]. Hajiaghayi et al. [31] showed that the problem is NP-complete even on planar graphs whereas it is solvable in polynomial time on bounded treewidth graphs. For the optimization version of Connected Maximum Cut, they proposed a polynomial time approximation scheme (PTAS) on planar graphs and more generally on bounded genus graphs and an $\Omega(1/\log n)$-approximation algorithm on general graphs.

Maximum Minimal Cut was considered in [30] and shown to be NP-complete on planar graphs. Recently, Chaourar proved that the problem can be solved in polynomial time on series parallel graphs [12] and graphs without $K_5 \setminus e$ as a minor [13].

As another related problem, Multi-Node Hubs was proposed by Saurabh and Zehavi [43]: Given a graph $G = (V, E)$ and two integers k, p, determine whether there is a connected cut of size at least k such that the size of the connected part is *exactly* p. They proved that Multi-Node Hubs is W[1]-hard with respect to p, but solvable in time $O^*(2^{2^{O(k)}})$. As an immediate corollary of their result, we can solve Connected Maximum Cut in time $O^*(2^{2^{O(k)}})$ by solving Multi-Node Hubs for each $0 \leq p \leq n$.

[2] The $O^*(\cdot)$ notation suppresses polynomial factors in the input size.

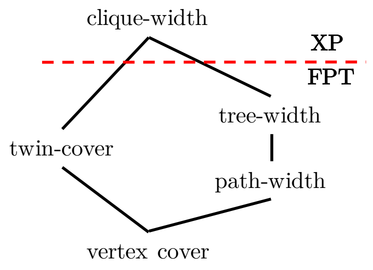


Figure 1 Graph parameters and the parameterized complexity of Maximum Cut, Connected Maximum Cut, and Maximum Minimal Cut. Connections between two parameters imply the above one is bounded by some function in the below one.

▶ **Proposition 1** ([43]). Connected Maximum Cut *can be solved in time* $O^*(2^{2^{O(k)}})$.

In this paper, we improve the running time in Proposition 1 by giving an $O^*(2^{O(k)})$-time algorithm for Connected Maximum Cut in Section 4.4.

2 Preliminaries

In this paper, we use the standard graph notations. Let $G = (V, E)$ be an undirected graph. For $V' \subseteq V$, we denote by $G[V']$ the subgraph of G induced by V'. We denote the open neighbourhood of v by $N(v)$ and the closed neighbourhood by $N[v]$.

A *cut* of G is a pair $(S, V \setminus S)$ for some subset $S \subseteq V$. Note that we allow S (and $V \setminus S$) to be empty. For simplicity, we sometimes denote a cut $(S, V \setminus S)$ by (S_1, S_2) where $S_1 = S$ and $S_2 = V \setminus S$. If the second part $V \setminus S$ of a cut is clear from the context, we may simply denote $(S, V \setminus S)$ by S. The *cutset* of S, denoted by $\delta(S)$, is the set of *cut edges* between S and $V \setminus S$. The size of a cut is defined as the number of edges in its cutset (i.e., $|\delta(S)|$). A cut S is *connected* if the subgraph induced by S is connected. We say that a cutset is *minimal* if there is no non-empty proper cutset of it. A cut is *minimal* if its cutset is minimal. It is well known that for every minimal cut S, $G[S]$ and $G[V \setminus S]$ are connected when G is connected [19]. If G has two or more connected component, every cutset of a minimal cut of G corresponds to a minimal cutset of its connected component. Therefore, throughout the paper, except in Theorem 24, we assume the input graph G is connected. Let p be a predicate. We define the function $[p]$ as follows: if p is true, then $[p] = 1$, otherwise $[p] = 0$.

2.1 Graph parameters

In this paper, we use the following graph parameters: clique-width $\mathrm{cw}(G)$, tree-width $\mathrm{tw}(G)$, path-width $\mathrm{pw}(G)$, twin-cover number $\mathrm{tc}(G)$, and vertex cover number $\mathrm{vc}(G)$. The definitions of them can be found in [14, 16, 23]. For clique-width, tree-width, path-width, twin-cover number, and vertex cover number, the following relations hold.

▶ **Proposition 2** ([6, 14, 23]). *For any graph G, the following inequalities hold:* $\mathrm{cw}(G) \leq 2^{\mathrm{tw}(G)+1} + 1$, $\mathrm{cw}(G) \leq \mathrm{pw}(G) + 1$, $\mathrm{tw}(G) \leq \mathrm{pw}(G) \leq \mathrm{vc}(G)$, $\mathrm{cw}(G) \leq 2^{\mathrm{tc}(G)} + \mathrm{tc}(G)$, *and* $\mathrm{tc}(G) \leq \mathrm{vc}(G)$.

From Proposition 2, we can illustrate the parameterized complexity of Maximum Cut, Connected Maximum Cut, and Maximum Minimal Cut associated with graph parameters in Figure 1.

3 Computational Complexity on Graph Classes

In this section, we prove that Connected Maximum Cut and Maximum Minimal Cut are NP-complete on planar bipartite graphs and split graphs.

3.1 Planar bipartite graphs

▶ **Theorem 3.** Connected Maximum Cut *is NP-complete on planar bipartite graphs.*

▶ **Theorem 4.** Maximum Minimal Cut *is NP-complete on planar bipartite subcubic graphs.*

Proof. We give a reduction from Maximum Minimal Cut on planar cubic graphs, which is known to be NP-complete [30]. Given a connected planar cubic graph $G = (V, E)$, we split each edge $e = \{u, w\} \in E$ by a vertex v_e, that is, we introduce a new vertex v_e and replace e by $\{u, v_e\}$ and $\{w, v_e\}$. Let $V_E = \{v_e \mid e \in E\}$ and $G' = (V \cup V_E, E')$ the reduced graph. Since we split each edge by a vertex and G is a planar cubic graph, G' is not only planar but also bipartite and subcubic. In the following, we show that there is a minimal cut of size at least k in G if and only if so is in G'. We can assume that $k > 2$.

Let (S_1, S_2) be a minimal cut of G. We construct a cut (S'_1, S'_2) of G' with $S_i \subseteq S'_i$ for $i = 1, 2$. For each edge $e \in E$, we add v_e to S'_2 if both endpoints of e are contained in S_2, and otherwise add v_e to S'_1. Recall that a cut is minimal if and only if both sides of the cut induce connected subgraphs. We claim that both $G'[S'_1]$ and $G'[S'_2]$ are connected. To see this, consider vertices $u, v \in S_1$. As $G[S_1]$ is connected, there is a path between u and v in $G[S_1]$. By the construction of S'_1, every vertex of the path is in S'_1 and for every edge e in the path, we have $v_e \in S'_1$. Therefore, there is a path between u and v in $G'[S'_1]$. Moreover, for every $v_e \in S'_1$, at least one endpoint of e is in S'_1. Hence, $G'[S'_1]$ is connected. Symmetrically, we can conclude that $G'[S'_2]$ is connected. Moreover, for each $e = \{u, w\}$ with $u \in S'_1$ and $w \in S'_2$, $\{v_e, w\}$ is a cut edge in G'. Therefore, (S'_1, S'_2) is a minimal cut of size at least k.

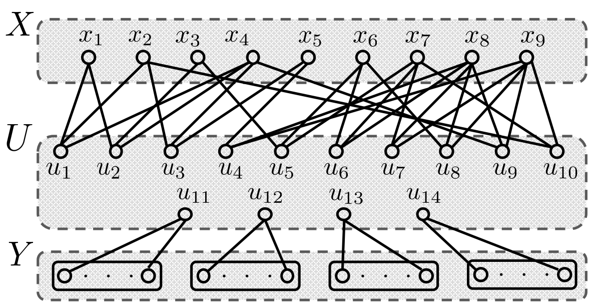
Conversely, we are given a minimal cut (S'_1, S'_2) of size k. We let $S_i = S'_1 \cap V$ for $i = 1, 2$. For each $e = \{u, v\}$, we can observe that $v_e \in S'_i$ if $u, w \in S'_i$ due to the connectivity of S'_i and $k > 2$. This means that an edge $\{u, v_e\}$ (or $\{w, v_e\}$) contributes to the cut if and only if exactly one of u and w is contained in S'_1 (and hence S_1), that is, the edge e contributes to the cut (S_1, S_2) in G. Therefore, the size of the cut (S_1, S_2) is at least k. Moreover, u and v are connected by a path through v_e in $G'[S'_i]$ if and only if u and v are contained in S_i and adjacent to each other in $G[S_i]$. Hence, $G[S_i]$ is connected for each $i = 1, 2$, and the theorem follows. ◀

3.2 Split graphs

▶ **Theorem 5.** Connected Maximum Cut *is NP-complete on split graphs.*

Proof. We reduce the following problem called Exact 3-cover, which is known to be NP-complete: Given a set $X = \{x_1, x_2, \dots, x_{3n}\}$ and a family $\mathcal{F} = \{F_1, F_2, \dots, F_m\}$, where each $F_i = \{x_{i_1}, x_{i_2}, x_{i_3}\}$ has three elements of X, the objective is to find a subfamily $\mathcal{F}' \subseteq \mathcal{F}$ such that every element in X is contained in exactly one of the subsets $\mathcal{F}'$. By making some copies of 3-element sets if necessary, we may assume that $|\{F \in \mathcal{F} \mid x \in F\}| \geq 3(n + 2)$ for each $x \in X$, which implies that m is sufficiently large compared to n.

Given an instance of Exact 3-cover with $|\{F \in \mathcal{F} \mid x \in F\}| \geq 3(n + 2)$ for each $x \in X$, we construct an instance of Connected Maximum Cut in a split graph as follows. We introduce m vertices $u_1, u_2, \dots, u_m$, where each u_i corresponds to F_i, and introduce $m - 2n$ vertices $u_{m+1}, u_{m+2}, \dots, u_{2(m-n)}$. Let $U := \{u_1, u_2, \dots, u_{2(m-n)}\}$. For



X
x_1 x_2 x_3 x_4 x_5 x_6 x_7 x_8 x_9
U
u_1 u_2 u_3 u_4 u_5 u_6 u_7 u_8 u_9 u_{10}
u_{11} u_{12} u_{13} u_{14}
Y

Figure 2 An instance of Connected Maximum Cut on split graphs reduced from an instance of Exact 3-cover where $X = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9\}$ and $\mathcal{F} = \{\{x_1, x_2, x_3\}, \{x_1, x_3, x_4\}, \{x_2, x_4, x_5\}, \{x_5, x_8, x_9\}, \{x_3, x_6, x_7\}, \{x_6, x_7, x_8\}, \{x_7, x_8, x_9\}, \{x_6, x_8, x_9\}, \{x_4, x_8, x_9\}, \{x_2, x_7, x_9\}\}$.

$i = m+1, m+2, \ldots, 2(m-n)$, introduce a vertex set Y_i of size M, where M is a sufficiently large integer compared to n (e.g. $M = 3n+1$). Now, we construct a graph $G = (U \cup X \cup Y, E)$, where $Y := \bigcup_{m+1 \le i \le m-2n} Y_i$, $E_U := \{\{u, u'\} \mid u, u' \in U, u \ne u'\}$, $E_X := \{\{u_i, x_j\} \mid 1 \le i \le m, 1 \le j \le 3n, x_j \in F_i\}$, $E_Y := \{\{u_i, y\} \mid m+1 \le i \le 2(m-n), y \in Y_i\}$, and $E := E_U \cup E_X \cup E_Y$. Then, G is a split graph in which U induces a clique and $X \cup Y$ is an independent set. We now show the following claim.

▷ Claim. The original instance of Exact 3-cover has a solution if and only if the obtained graph G has a connected cut of size at least $(m-n)^2 + 3m - 3n + (m-2n)M$.

Proof. Suppose that the original instance of Exact 3-cover has a solution $\mathcal{F}'$. Then $S := \{u_i \mid F_i \in \mathcal{F}'\} \cup \{u_i \mid m+1 \le i \le 2(m-n)\} \cup X$ is a desired connected cut, because $|\delta(S) \cap E_U| = (m-n)^2$, $|\delta(S) \cap E_X| = \sum_{i=1}^{m} |F_i| - |X| = 3m - 3n$, and $|\delta(S) \cap E_Y| = (m-2n)M$.

Conversely, suppose that the obtained instance of Connected Maximum Cut has a connected cut S such that $|\delta(S)| \ge (m-n)^2 + 3m - 3n + (m-2n)M$. Since $|\delta(S) \cap E_U| \le (m-n)^2$, $|\delta(S) \cap E_X| \le 3m$, and $|\delta(S) \cap E_Y| \le |S \cap \{u_{m+1}, \ldots, u_{2(m-n)}\}| \cdot M$, we obtain $|S \cap \{u_{m+1}, \ldots, u_{2(m-n)}\}| = m - 2n$, that is, $\{u_{m+1}, \ldots, u_{2(m-n)}\} \subseteq S$. Let $t = |S \cap \{u_1, \ldots, u_m\}|$, $X_0 = \{x \in X \mid N(x) \cap S = \emptyset\}$ the vertices in X that has no neighbor in S, $X_{\text{all}} = \{x \in X \mid N(x) \subseteq S\}$ the vertices in X whose neighbor is entirely included in S, and $X_{\text{part}} = X \setminus (X_0 \cup X_{\text{all}})$ all the other vertices in X. Recall that every element in X is contained in at least $3(n+2)$ subsets of $\mathcal{F}$. Then, since $|\delta(S) \cap E_U| = (m-t)(m-2n+t) = (m-n)^2 - (t-n)^2$, $|\delta(S) \cap E_X| \le |E_X| - |X_{\text{part}}| - |\delta(X_0)| \le 3m - (3n - |X_{\text{all}}| - |X_0|) - 3(n+2)|X_0|$, $|\delta(S) \cap E_Y| \le (m-2n)M$, and $|\delta(S)| \ge (m-n)^2 + 3m - 3n + (m-2n)M$, we obtain

$$|X_{\text{all}}| - (3n+5)|X_0| - (t-n)^2 \ge 0. \tag{1}$$

By counting the number of edges between $S \cap \{u_1, u_2, \ldots, u_m\}$ and X, we obtain $3t \ge |\delta(X_{\text{all}})| \ge 3(n+2)|X_{\text{all}}|$, which shows that $t \ge (n+2)|X_{\text{all}}|$. If $|X_{\text{all}}| \ge 1$, then $t \ge (n+2)|X_{\text{all}}| \ge n + 2|X_{\text{all}}|$, and hence $|X_{\text{all}}| - 3(n+5)|X_0| - (t-n)^2 \le |X_{\text{all}}| - (2|X_{\text{all}}|)^2 < 0$, which contradicts (1). Thus, we obtain $|X_{\text{all}}| = 0$, and hence we have $t = n$ and $X_0 = \emptyset$ by (1). Therefore, $\mathcal{F}' := \{F_i \mid 1 \le i \in m,\ u_i \in S\}$ satisfies that $|\mathcal{F}'| = n$ and $\bigcup_{F \in \mathcal{F}'} F = X$. This shows that $\mathcal{F}'$ is a solution of the original instance of Exact 3-cover. ◁

This shows that Exact 3-cover is reduced to Connected Maximum Cut in split graphs, which completes the proof. ◀

▶ **Theorem 6.** Maximum Minimal Cut *is NP-complete on split graphs.*

4 Parameterized Complexity

4.1 Tree-width

In this section, we give FPT algorithms for Connected Maximum Cut and Maximum Minimal Cut parameterized by tree-width. In particular, we design $O^*(c^{\mathrm{tw}})$-time algorithms where c is some constant.

4.1.1 $O^*(\mathrm{tw}^{O(\mathrm{tw})})$-algorithm

We design an $O^*(\mathrm{tw}^{O(\mathrm{tw})})$-algorithm for Maximum Minimal Cut. To do this, we consider a slightly different problem, called Maximum Minimal s-t Cut: Given a graph $G = (V, E)$, an integer k and two vertices $s, t \in V$, determine whether there is a cut (S_1, S_2) of size at least k in G such that $s \in S_1$, $t \in S_2$ and (S_1, S_2) is minimal, that is, both $G[S_1]$ and $G[S_2]$ are connected. If we can solve Maximum Minimal s-t Cut in time $O^*(\mathrm{tw}^{O(\mathrm{tw})})$, we can also solve Maximum Minimal Cut in the same running time up to a polynomial factor in n since it suffices to compute Maximum Minimal s-t Cut for each pair of s and t.

Our algorithm is based on standard dynamic programming on a tree decomposition. This algorithm outputs a maximum minimal cut (S_1, S_2). Basically, the algorithm is almost the same as an $O^*(2^{\mathrm{tw}})$-algorithm for Max Cut in [7] except for keeping the connectivity of a cut. In other words, for each vertex, we label either **1** or **2**, which represent a vertex is assigned to S_1 or S_2. To keep track of the connectivity, for each bag X_i, we consider two partitions $\mathcal{S}_1$ and $\mathcal{S}_2$ of $S_1 \cap X_i$ and $S_2 \cap X_i$, respectively.

▶ **Theorem 7.** *Given a tree decomposition of width* tw *of G,* Maximum Minimal Cut *and* Maximum Minimal *s-t* Cut *can be solved in time $O^*(\mathrm{tw}^{O(\mathrm{tw})})$.*

The algorithm in Theorem 7 is applicable to Connected Maximum s-t Cut and Connected Maximum Cut as well.

▶ **Theorem 8.** *Give a tree decomposition of width* tw*,* Connected Maximum *s-t* Cut *and* Connected Maximum Cut *are solvable in time $O^*(\mathrm{tw}^{O(\mathrm{tw})})$.*

The dynamic programming algorithms in Theorems 7, 8 can be seen as ones for *connectivity problems* such as finding a Hamiltonian cycle, a feedback vertex set, and a Steiner tree. For such problems, we can improve the running time $\mathrm{tw}^{O(\mathrm{tw})}$ to $2^{O(\mathrm{tw})}$ using two techniques called the *rank-based approach* due to Bodlaender et al. [3] and the *cut & count technique* due to Cygan et al. [17]. In the next two subsections, we improve the running time of the algorithms described in this section using these techniques.

4.1.2 Rank-based approach

In this subsection, we provide faster $2^{O(\mathrm{tw})}$-time deterministic algorithms parameterized by tree-width. To show this, we use the rank-based approach proposed by Bodlaender et al. [3]. The key idea of the rank-based approach is to keep track of *small* representative sets of size $2^{O(\mathrm{tw})}$ that capture partial solutions of an optimal solution instead of $\mathrm{tw}^{O(\mathrm{tw})}$ partitions. Indeed, we can compute small representative sets within the claimed running time using reduce algorithm [3].

▶ **Theorem 9.** *Given a tree decomposition of width* tw*, there are $O^*((1 + 2^{\omega+1})^{\mathrm{tw}})$-time deterministic algorithms for* Connected Maximum *s-t* Cut *and* Connected Maximum Cut.

▶ **Theorem 10.** *Given a tree decomposition of width* tw*, there are $O^*(2^{(\omega+2)\text{tw}})$-time deterministic algorithms for* Maximum Minimal *s*-*t* Cut *and* Maximum Minimal Cut.

4.1.3 Cut & Count

In this subsection, we design much faster randomized algorithms by using Cut & Count, which is the framework for solving the connectivity problems faster [17]. In Cut & Count, we count the number of *relaxed* solutions modulo 2 on a tree decomposition and determine whether there exists a connected solution by cancellation tricks.

▶ **Theorem 11.** *Given a tree decomposition of width* tw*, there is a Monte-Carlo algorithm that solves* Maximum Minimal Cut *and* Maximum Minimal *s*-*t* Cut *in time $O^*(4^{\text{tw}})$. It cannot give false positives and may give false negatives with probability at most 1/2.*

▶ **Theorem 12.** *Given a tree decomposition of width* tw*, there is a Monte-Carlo algorithm that solves* Connected Maximum Cut *and* Connected Maximum *s*-*t* Cut *in time $O^*(3^{\text{tw}})$. It cannot give false positives and may give false negatives with probability at most 1/2.*

4.2 Clique-width

In this section, we design XP algorithms for both Connected Maximum Cut and Maximum Minimal Cut when parameterized by clique-width. The algorithms are analogous to the dynamic programming algorithm for Maximum Cut given by Fomin et al. [22], but we need to carefully control the connectivity information in partial solutions.

Suppose that the clique-width of G is w. Then, G can be constructed by the four operations: *creation*, *disjoint union*, *joining*, and *relabeling* (see e.g., [14]). This construction naturally defines a tree expressing a sequence of operations. This tree is called a *w-expression tree* of G and used for describing dynamic programming algorithms for many problems based on clique-width. Here, we rather use a different graph parameter and its associated decomposition closely related to clique-width. We believe that this decomposition is more suitable to describe our dynamic programming.

▶ **Definition 13.** *Let $X \subseteq V(G)$. We say that $M \subseteq X$ is a* twin-set *of X if for any $v \in V(G) \setminus X$, either $M \subseteq N(v)$ or $M \cap N(v) = \emptyset$ holds. A twin-set M is called a* twin-class *of X if it is maximal subject to being a twin-set of X. X can be partitioned into twin-classes of X.*

▶ **Definition 14.** *Let w be an integer. We say that $X \subseteq V(G)$ is a w-*module *of G if X can be partitioned into w twin-classes $\{X_1, X_2, \ldots, X_w\}$. A* decomposition tree *of G is a pair of a rooted binary tree T and a bijection ϕ from the set of leaves of T to $V(G)$. For each node v of T, we denote by L_v the set of leaves, each of which is either v or a descendant of v. The* width *of a decomposition tree (T, ϕ) of G is the minimum w such that for every node v in T, the set $\bigcup_{l \in L_v} \phi(l)$ is a w_v-module of G with $w_v \le w$. The* module-width *of G is the minimum t such that there is a decomposition tree of G of width w.*

Rao [41] proved that clique-width and module-width are linearly related to each other. Let $\text{cw}(G)$ and $\text{mw}(G)$ be the clique-width and the module-width of G, respectively. We note that a similar terminology "modular-width" has been used in many researches, but module-width used in this paper is different from it.

▶ **Theorem 15** ([41])**.** *For every graph G, $\text{mw}(G) \le \text{cw}(G) \le 2\text{mw}(G)$.*

Moreover, given a w-expression tree of G, we can in time $O(n^2)$ compute a decomposition tree (T, ϕ) of G of width at most w and $w_v \le w$ twin-classes of $\bigcup_{l \in L_v} \phi(l)$ for each node v in T [10].

Fix a decomposition tree (T, f) of G whose width is w. Our dynamic programming algorithm runs over the nodes of the decomposition tree in a bottom-up manner. For each node v in T, we let $\{X^v_1, X^v_2, \dots, X^v_{w_v}\}$ be the twin-classes of $\bigcup_{l \in L_v} \phi(l)$. From now on, we abuse the notation to denote $\bigcup_{l \in L_v} \phi(l)$ simply by L_v. A tuple of $4w_v$ integers $t = (p_1, \overline{p}_1, p_2, \overline{p}_2, \dots, p_{w_v}, \overline{p}_{w_v}, c_1, \overline{c}_1, c_2, \overline{c}_2, \dots, c_{w_v}, \overline{c}_{w_v})$ is *valid* for v if it holds that $0 \le p_i, \overline{p}_i \le |X^v_i|$ with $p_i + \overline{p}_i = |X^v_i|$ and $c_i, \overline{c}_i \in \{0, 1\}$ for each $1 \le i \le w_v$. For a valid tuple t for v, we say that a cut $(S, L_v \setminus S)$ of $G[L_v]$ is *t-legitimate* if for each $1 \le i \le w_v$, it satisfies the following conditions:

- $p_i = |S \cap X^v_i|$,
- $\overline{p}_i = |(L_v \setminus S) \cap X^v_i|$,
- $G[S \cap X^v_i]$ is connected if $c_i = 1$, and
- $G[(L_v \setminus S) \cap X^v_i]$ is connected if $\overline{c}_i = 1$.

The size of a t-legitimate cut is defined accordingly. In this section, we allow each side of a cut to be empty and the empty graph is considered to be connected. Our algorithm computes the value $\mathrm{mc}(v, t)$ that is the maximum size of a t-legitimate cut for each valid tuple t and for each node v in the decomposition tree.

Leaves (Base step):

For each valid tuple t for a leaf v, $\mathrm{mc}(v, t) = 0$. Note that there is only one twin-class $X^v_1 = \{v\}$ for v in this case.

Internal nodes (Induction step):

Let v be an internal node of T and let a and b be the children of v in T. Consider twin-classes $\mathcal{X}^v = \{X^v_1, X^v_2, \dots, X^v_{w_v}\}$, $\mathcal{X}^a = \{X^a_1, X^a_2, \dots, X^a_{w_a}\}$, and $\mathcal{X}^b = \{X^b_1, X^b_2, \dots, X^b_{w_b}\}$ of L_v, L_a, and L_b, respectively. Note that $\mathcal{X}^a \cup \mathcal{X}^b$ is a partition of L_v.

▶ **Observation 1.** *$\mathcal{X}^v$ is a partition of L_v coarser than $\mathcal{X}^a \cup \mathcal{X}^b$.*

To see this, consider an arbitrary twin-class X^a_i of L_a. By the definition of twin-sets, for every $z \in V(G) \setminus L_a$, either $X^a_i \subseteq N(z)$ or $X^a_i \cap N(z) = \emptyset$ holds. Since $V(G) \setminus L_v \subseteq V(G) \setminus L_a$, X^a_i is also a twin-set of L_v, which implies X^a_i is included in some twin-class X^v_j of L_v. This argument indeed holds for twin-classes of L_b. Therefore, we have the above observation.

The intuition of our recurrence is as follows. By Observation 1, every twin-class of L_v can be obtained by merging some twin-classes of L_a and of L_b. This means that every t_v-legitimate cut of $G[L_v]$ for a valid tuple t_v for v can be obtained from some t_a-legitimate cut and t_b-legitimate cut for valid tuples for a and b, respectively. Moreover, for every pair of twin-classes X^a_i of L_a and X^b_j of L_b, either there are no edges between them or every vertex in X^a_i is adjacent to every vertex in X^b_j as X^a_i is a twin-set of L_v. Therefore, the number of edges in the cutset of a cut $(S, L_v \setminus S)$ between X^a_i and X^b_j depends only on the cardinality of $X^a_i \cap S$ and $X^b_j \cap S$ rather than actual cuts $(S \cap X^a_i, (L_a \setminus S) \cap X^a_i)$ and $(S \cap X^b_i, (L_b \setminus S) \cap X^b_i)$.

Now, we formally describe this idea. Let X^v be a twin-class of L_v. We denote by $I_a(X^v)$ (resp. $I_b(X^v)$) the set of indices i such that X^a_i (resp. X^b_i) is included in X^v and by $\mathcal{X}^a(X^v)$ (resp. $\mathcal{X}^b(X^v)$) the set $\{X^a_i : i \in I_a(X^v)\}$ (resp. $\{X^b_i : i \in I_b(X^v)\}$). For $X^a \in \mathcal{X}^a(X^v)$ and $X^b \in \mathcal{X}^a(X^v)$, we say that X^a is adjacent to X^b if every vertex in X^a is adjacent to every vertex in X^b and otherwise X^a is not adjacent to X^b. This adjacency relation naturally defines a bipartite graph whose vertex set is $\mathcal{X}^a(X^v) \cup \mathcal{X}^b(X^v)$. We say that a subset of twin-classes of $\mathcal{X}^a(X^v) \cup \mathcal{X}^b(X^v)$ is *non-trivially connected* if it induces a connected bipartite graph with at least twin-classes. Let $S \subseteq X^v$. To make $G[S]$ (and $G[X^v \setminus S]$) connected, the following observation is useful.

▶ **Observation 2.** *Suppose $S \subseteq X^v$ has a non-empty intersection with at least two twin-classes of $\mathcal{X}^a(X^v) \cup \mathcal{X}^b(X^v)$. Then, $G[S]$ is connected if and only if the twin-classes having a non-empty intersection with S are non-trivially connected.*

This observation immediately follows from the fact that every vertex in a twin-class is adjacent to every vertex in an adjacent twin-class and is not adjacent to every vertex in a non-adjacent twin-class.

Let $t_v = (p^v_1, \overline{p}^v_1, \ldots, p^v_{w_v}, \overline{p}^v_{w_v}, c^v_1, \overline{c}^v_2, \ldots, c^v_{w_v}, \overline{c}^v_{w_v})$ be a valid tuple for v. For notational convenience, we use $\mathbf{p}^v$ to denote $(p^v_1, \overline{p}^v_1, \ldots, p^v_{w_v}, \overline{p}^v_{w_v})$ and $\mathbf{c}^v$ to denote $(c^v_1, \overline{c}^v_2, \ldots, c^v_{w_v}, \overline{c}^v_{w_v})$ for each node v in T. For valid tuples $t_a = (\mathbf{p}^a, \mathbf{c}^a)$ for a and $t_b = (\mathbf{p}^b, \mathbf{c}^b)$ for b, we say that t_v *is consistent with the pair* (t_a, t_b) if for each $1 \le i \le w_v$,

C1 $p^v_i = \sum_{j \in I_a(X^v_i)} p^a_j + \sum_{j \in I_b(X^v_i)} p^b_j$;
C2 $\overline{p}^v_i = \sum_{j \in I_a(X^v_i)} \overline{p}^a_j + \sum_{j \in I_b(X^v_i)} \overline{p}^b_j$;
C3 if $c^v_i = 1$, either (1) $\{X^a_j : j \in I_a(X^v), p^a_j > 0\} \cup \{X^b_j : j \in I_b(X^v), p^b_j > 0\}$ is non-trivially connected or (2) exactly one of $\{p^s_j : s \in \{a, b\}, 1 \le j \le w_s\}$ is positive, say p^s_j, and $c^s_j = 1$;
C4 if $\overline{c}^v_i = 1$, either (1) $\{X^a_j : j \in I_a(X^v), \overline{p}^a_j > 0\} \cup \{X^b_j : j \in I_b(X^v), \overline{p}^b_j > 0\}$ is non-trivially connected or (2) exactly one of $\{\overline{p}^s_j : s \in \{a, b\}, 1 \le j \le w_s\}$ is positive, say $\overline{p}^s_j$, and $\overline{c}^s_j = 1$.

▶ **Lemma 16.**

$$\mathrm{mc}(v, t_v) = \max_{t_a, t_b} \left(\mathrm{mc}(a, t_a) + \mathrm{mc}(b, t_b) + \sum_{\substack{X^a_i \in \mathcal{X}^a, X^b_j \in \mathcal{X}^b \\ X^a_i, X^b_j : adjacent}} (p^a_i \overline{p}^b_j + p^b_j \overline{p}^a_i) \right),$$

where the maximum is taken over all consistent pairs (t_a, t_b).

Proof. We first show that the left-hand side is at most the right-hand side. Suppose $(S, L_v \setminus S)$ be a t_v-legitimate cut of $G[L_v]$ whose size is equal to $\mathrm{mc}(v, t_v)$. Let $S_a = S \cap L_a$ and $S_b = S \cap L_b$. We claim that $(S_a, L_a \setminus S_a)$ is a t_a-legitimate cut of $G[L_a]$ for some valid tuple t_a for a. This is obvious since we set $p^a_i = |S_a \cap X^a_i|$, $\overline{p}^a_i = |(L_a \setminus S_a) \cap X^a_i|$, $c^a_i = 1$ if $G[S_a \cap X^a_i]$ is connected, and $c^a_i = 1$ if $G[(L_a \setminus S_a) \cap X^a_i]$ is connected, which yields a valid tuple t_a for a. We also conclude that $(S_b, L_b \setminus S_b)$ is a t_b-legitimate cut of $G[L_b]$ for some valid tuple t_b for b. Moreover, the number of cut edges between twin-class X^a_i of L_a and twin-class X^b_j of L_b is $|S_a \cap X^a_i| \cdot |(L_b \setminus S_b) \cap X^b_j| + |S_b \cap X^b_j| \cdot |(L_b \setminus S_a) \cap X^a_i| = p^a_i \overline{p}^b_j + p^b_j \overline{p}^a_i$ if X^a_i and X^b_j is adjacent, zero otherwise. Therefore, the left-hand side is at most the right-hand side.

To show the converse direction, suppose $(S_a, L_a \setminus S_a)$ is a t_a-legitimate cut of $G[L_a]$ and $(S_b, L_b \setminus S_b)$ is a t_b-legitimate cut of $G[L_b]$, where t_v is consistent with (t_a, t_b) and the sizes of the cuts are $\mathrm{mc}(a, t_a)$ and $\mathrm{mc}(b, t_b)$, respectively. We claim that $(S_a \cup S_b, L_v \setminus (S_a \cup S_b))$ is a t_v-legitimate cut of $G[L_v]$. Since t_v is consistent with (t_a, t_b), for each $1 \le i \le w_v$, we have $p^v_i = \sum_{j \in I_a(X^v_i)} p^a_j + \sum_{j \in I_b(X^v_i)} p^b_j = \sum_{1 \le j \le w_a} |S_a \cap X^i_v| + \sum_{1 \le j \le w_b} |S_b \cap X^i_v| = |(S_a \cup S_b) \cap X^i_v|$. Symmetrically, we have $\overline{p}^i = |(L_v \setminus (S_a \cup S_b)) \cap X^v_i|$. If $c^v_i = 1$, by condition C3 of the consistency, either (1) $\{X^a_j : j \in I_a(X^v), p^a_j > 0\} \cup \{X^b_j : j \in I_b(X^v), p^b_j > 0\}$ is non-trivially connected or (2) exactly one of $\{p^s_j : s \in \{a, b\}, 1 \leq j \leq w_s\}$ is positive, say p^s_j, and $c^s_j = 1$. If (1) holds, by Observation 2, $G[(S_a \cap S_b) \cap X^i_v]$ is connected. Otherwise, as $c^s_j = 1$, $G[S_s \cap X^i_v] = G[(S_a \cup S_b) \cap X^v_i]$ is also connected. By a symmetric argument, we conclude that $G[(L_v \setminus (S_a \cup S_b)) \cap X^i_v]$ is connected if $\overline{c}^v_i = 1$. Therefore the cut $(S_a \cup S_b, L_v \setminus (S_a \cup S_b))$ is t_v-legitimate. Since the cut edges between two twin-classes of L_a is counted by $\mathrm{mc}(a, t_a)$ and those between two twin-classes of L_v is counted by $\mathrm{mc}(b, t_b)$. Similar to the forward direction, the number of cut edges between a twin-class of L_a and a twin-class of L_b can be counted by the third term in the right-hand side of the equality. Hence, the left-hand side is at least right-hand side. ◀

▶ **Theorem 17.** Connected Maximum Cut *and* Maximum Minimal Cut *can be computed in time $n^{O(w)}$ provided that a w-expression tree of G is given as input.*

Proof. From a w-expression tree of G, we can obtain a decomposition tree (T, ϕ) of width at most w in $O(n^2)$ time using Rao's algorithm [41]. Based on this decomposition, we evaluate the recurrence in Lemma 16 in a bottom-up manner. The number of valid tuples for each node of T is at most $4^w n^w$. For each internal node v and for each valid tuple t_v for v, we can compute $\mathrm{mc}(v, t_v)$ in $(4^w n^w)^2 n^{O(1)}$ time. Overall, the running time of our algorithm is $n^{O(w)}$. Let r be the root of T. For Connected Maximum Cut, by the definition of legitimate cuts, we should take the maximum value among $\mathrm{mc}(r, (i, n-i, 1, j))$ for $1 \leq i < n$ and $j \in \{0, 1\}$. Note that as L_v has only one twin-class, the length of valid tuples is exactly four. For Maximum Minimal Cut, we should take the maximum value among $\mathrm{mc}(r, (i, n-i, 1, 1))$ for $1 \leq i < n$. ◀

Since there is an algorithm that, given a graph G and an integer k, either conclude that the clique-width of G is more than k or find a $(2^{k-1} - 1)$-expression tree of G in time $O(n^3)$ [33, 39, 38], Maximum Minimal Cut and Connected Maximum Cut are XP parameterized by the clique-width of the input graph.

4.3 Twin-cover

Maximum Cut is FPT when parameterized by twin-cover number [23]. In this section, we show that Connected Maximum Cut and Maximum Minimal Cut are also FPT when parameterized by twin-cover number.

▶ **Theorem 18.** Connected Maximum Cut *can be solved in time $O^*(2^{2^{\mathrm{tc}}+\mathrm{tc}})$.*

Proof. We first compute a minimum twin-cover X of $G = (V, E)$ in time $O^*(1.2738^{\mathrm{tc}})$ [23]. Now, we have a twin-cover X of size tc. Recall that $G[V \setminus X]$ consists of vertex disjoint cliques and for each $u, v \in Z$ in a clique Z of $G[V \setminus X]$, $N(u) \cap X = N(v) \cap X$.

We iterate over all possible subsets X' of X and compute the size of a maximum cut $(S, V \setminus S)$ of G with $S \cap X = X'$.

If $X' = \emptyset$, exactly one of the cliques of $G[V \setminus X]$ intersects S as $G[S]$ is connected. Thus, we can compute a maximum cut by finding a maximum cut for each clique of $G[V \setminus X]$, which can be done in polynomial time.

Suppose otherwise that $X' \neq \emptyset$. We define a *type* of each clique Z of $G[V \setminus X]$. The type of Z, denoted by $T(Z)$, is $N(Z) \cap X$. Note that there are at most $2^{\mathrm{tc}} - 1$ types of cliques in $G[V \setminus X]$.

For each type of cliques, we guess that S has an intersection with this type of cliques. There are at most $2^{2^{\mathrm{tc}}-1}$ possible combinations of types of cliques. Let $\mathcal{T}$ be the set of types in $G[V \setminus X]$. For each guess $\mathcal{T}' \subseteq \mathcal{T}$, we try to find a maximum cut $(S, V \setminus S)$ such that $G[S]$ is connected, $S \cap X = X'$, for each $T \in \mathcal{T}'$, at least one of the cliques of type T has an intersection with S, and for each $T \notin \mathcal{T}'$, every clique of type T has no intersection with S. We can easily check if $G[S]$ will be connected as S contains a vertex of a clique of type $T \in \mathcal{T}'$. Consider a clique Z of type $T(Z) = X'' \subseteq X$. Since every vertex in Z has the same neighborhood in X, we can determine the number of cut edges incident to Z from the cardinality of $S \cap Z$. More specifically, if $|S \cap Z| = p$, the number of cut edges incident to Z is equal to $p(|Z| - p) + p|X'' \cap (X \setminus X')| + (|Z| - p)|X'' \cap X'|$. Moreover, we can independently maximize the number of cut edges incident to Z for each clique Z of $G[V \setminus X]$.

Overall, for each $X' \subseteq X$ and for each set of types $\mathcal{T}'$, we can compute a maximum connected cut with respect to X' and $\mathcal{T}'$ in polynomial time. Therefore, the total running time is bounded by $O^*(2^{2^{\mathrm{tc}}+\mathrm{tc}})$. ◀

▶ **Theorem 19.** Maximum Minimal Cut *can be solved in time* $O^*(2^{\mathrm{tc}}3^{2^{\mathrm{tc}}})$.

4.4 Solution size

In this section, we give FPT algorithms parameterized by the solution size for Connected Maximum Cut and Maximum Minimal Cut. To show this, we use the following theorem.

▶ **Theorem 20** ([2]). *The Cartesian product $C_k \times K_2$ of a k-circuit with K_2 is called a* k-prism. *If G contains no k-prism as a minor,* $\mathrm{tw}(G) = O(k^2)$.

Then we have the following theorem.

▶ **Theorem 21.** Connected Maximum Cut *and* Maximum Minimal Cut *can be solved in time $O^*(2^{O(k^2)})$ where k is the solution size.*

Proof. We first determine whether the tree-width of G is $O(k^2)$ in time $O^*(2^{O(k)})$ by using the algorithm in [5]. If $\mathrm{tw}(G) = O(k^2)$, the algorithm in [5] outputs a tree decomposition of width $O(k^2)$. Thus, we apply the dynamic programming algorithms based on tree decompositions described in Section 4.1, and the running time is $O^*(2^{O(k^2)})$. Otherwise, we can conclude that G has a minimal cut (and also a connected cut) of size at least k. To see this, consider a k-prism minor of G. Then, we take k "middle edges" corresponding to K_2 in the k-prism minor and add some edges to make these edges form a cutset of some minimal cut of G. The size of such a cut is at least k and hence G has a minimal cut and a connected cut of size at least k. ◀

For Connected Maximum Cut, we can further improve the running time by giving an $O^*(9^k)$-time algorithm.

In [21], Fellows et al. proposed a "Win/Win" algorithm that outputs in linear time either a spanning tree of G having at least k leaves, or a path decomposition of G of width at most $2k$. If G has such a spanning tree, we can construct a cut $(S, V \setminus S)$ of size at least k by taking the internal vertices of the tree for S. Clearly, $G[S]$ is connected, and hence we are done in this case. Otherwise, we have a path decomposition of width at most $2k$. Thus, we can compute Connected Maximum Cut on such a path decomposition by using an $O^*(3^{\mathrm{tw}})$-algorithm in Section 4.1.

▶ **Theorem 22.** *There is a Monte-Carlo algorithm that solves* Connected Maximum Cut *in time $O^*(9^k)$. It cannot give false positives and may give false negatives with probability at most 1/2.*

Also, using the rank-based algorithm in Theorem 9, we obtain an $O^*(38.2^k)$-time deterministic algorithm for Connected Maximum Cut. Note that our rank-based algorithm in Theorem 9 runs in time $O^*((1+2^\omega)^{\mathrm{pw}})$ on a path decomposition and $(1+2^\omega)^2 < 38.2$, where $\omega < 2.3727$ is the exponent of matrix multiplication.

▶ **Theorem 23.** *There is an $O^*(38.2^k)$-time deterministic algorithm for* Connected Maximum Cut.

As for kernelization, it is not hard to see that Connected Maximum Cut and Maximum Minimal Cut do not admit a polynomial kernelization unless NP $\subseteq$ coNP/poly since both problems are trivially OR-compositional [4]; at least one of graphs $G_1, G_2, \ldots G_t$ have a connected/minimal cut of size at least k if and only if their disjoint union $G_1 \cup G_2 \cup \cdots \cup G_t$ has.

▶ **Theorem 24.** *Unless* NP $\subseteq$ coNP/poly*,* Maximum Minimal Cut *and* Connected Maximum Cut *admit no polynomial kernel parameterized by the solution size.*

5 Conclusion and Remark

In this paper, we studied two variants of Max Cut, called Connected Maximum Cut and Maximum Minimal Cut. We showed that both problems are NP-complete even on planar bipartite graphs and split graphs. For the parameterized complexity, we gave FPT algorithms parameterized by tree-width, twin-cover number, and the solution size, respectively. Moreover, we designed XP-algorithms parameterized by clique-width.

Finally, we mention our problems on weighted graphs. It is not hard to see that Connected Maximum Cut and Maximum Minimal Cut remain to be FPT with respect to tree-width. However, our results with respect to clique-width and twin-cover number would not be extended to weighted graphs since both problems are NP-hard on 0-1 edge-weighted complete graphs.

References

1 C. Bazgan, L. Brankovic, K. Casel, H. Fernau, K. Jansen, K.-M. Klein, M. Lampis, M. Liedloff, J. Monnot, and V. T. Paschos. The many facets of upper domination. *Theoretical Computer Science*, 717:2–25, 2018.

2 E. Birmelé, J. A. Bondy, and B. A. Reed. *Brambles, Prisms and Grids*, pages 37–44. Birkhäuser Basel, Basel, 2007.

3 H. L. Bodlaender, M. Cygan, S. Kratsch, and J. Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Information and Computation*, 243:86–111, 2015.

4 H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.

5 H. L. Bodlaender, P. G. Drange, M. S. Dregi, F. V. Fomin, D. Lokshtanov, and M. Pilipczuk. A $c^k n$ 5-Approximation Algorithm for Treewidth. *SIAM Journal on Computing*, 45(2):317–378, 2016.

6 H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks. Approximating Treewidth, Pathwidth, Frontsize, and Shortest Elimination Tree. *Journal of Algorithms*, 18(2):238–255, 1995.

7 H. L. Bodlaender and K. Jansen. On the Complexity of the Maximum Cut Problem. *Nordic Journal of Computing*, 7(1):14–31, 2000.

8 N. Boria, F. D. Croce, and V. T. Paschos. On the max min vertex cover problem. *Discrete Applied Mathematics*, 196:62–71, 2015.

9 A. Boyacı, T. Ekim, and M. Shalom. A polynomial-time algorithm for the maximum cardinality cut problem in proper interval graphs. *Information Processing Letters*, 121:29–33, 2017.

10 B.-M. Bui-Xuan, O. Suchý, J. A. Telle, and M. Vatshelle. Feedback vertex set on graphs of low clique-width. *European Journal of Combinatorics*, 34(3):666–679, 2013.

11 R. Carvajal, M. Constantino, M. Goycoolea, J. P. Vielma, and A. Weintraub. Imposing Connectivity Constraints in Forest Planning Models. *Operations Research*, 61(4):824–836, 2013.

12 B. Chaourar. A Linear Time Algorithm for a Variant of the MAX CUT Problem in Series Parallel Graphs. *Advances in Operations Research*, pages 1267108:1–1267108:4, 2017.

13 B. Chaourar. Connected max cut is polynomial for graphs without $K_5 \setminus e$ as a minor. *CoRR*, abs/1903.12641, 2019.

14 B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1):77–114, 2000.

15 M. Cygan. Deterministic Parameterized Connected Vertex Cover. In *SWAT 2012*, pages 95–106, 2012.

16 M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer International Publishing, 2015.

17 M. Cygan, J. Nederlof, M. Pilipczuk, M. Pilipczuk, J. M. M. van Rooij, and J. O. Wojtaszczyk. Solving Connectivity Problems Parameterized by Treewidth in Single Exponential Time. In *FOCS 2011*, pages 150–159, 2011.

18 M. Demange. A Note on the Approximation of a Minimum-Weight Maximal Independent Set. *Computational Optimization and Applications*, 14(1):157–169, 1999.

19 R. Diestel. *Graph Theory, 4th Edition*, volume 173 of *Graduate texts in mathematics*. Springer, 2012.

20 J. Díaz and M. Kamiński. MAX-CUT and MAX-BISECTION are NP-hard on unit disk graphs. *Theoretical Computer Science*, 377(1):271–276, 2007.

21 M. R. Fellows, D. Lokshtanov, N. Misra, M. Mnich, F. Rosamond, and S. Saurabh. The Complexity Ecology of Parameters: An Illustration Using Bounded Max Leaf Number. *Theory of Computing Systems*, 45(4):822–848, 2009.

22 F. V. Fomin, P. Golovach, D. Lokshtanov, and S. Saurabh. Almost Optimal Lower Bounds for Problems Parameterized by Clique-Width. *SIAM Journal on Computing*, 43(5):1541–1563, 2014.

23 R. Ganian. Improving Vertex Cover as a Graph Parameter. *Discrete Mathematics and Theoretical Computer Science*, 17(2):77–100, 2015.

24 M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1979.

25 M. X. Goemans and D. P. Williamson. Improved Approximation Algorithms for Maximum Cut and Satisfiability Problems Using Semidefinite Programming. *Journal of the ACM*, 42(6):1115–1145, 1995.

26 V. Grimm, T. Kleinert, F. Liers, M. Schmidt, and G. Zöttl. Optimal price zones of electricity markets: a mixed-integer multilevel model and global solution approaches. *Optimization Methods and Software*, 34(2):406–436, 2019.

27 S. Guha and S. Khuller. Approximation Algorithms for Connected Dominating Sets. *Algorithmica*, 20(4):374–387, 1998.

28 V. Guruswami. Maximum cut on line and total graphs. *Discrete Applied Mathematics*, 92(2):217–221, 1999.

29 F. Hadlock. Finding a Maximum Cut of a Planar Graph in Polynomial Time. *SIAM Journal on Computing*, 4(3):221–225, 1975.

30 D. J. Haglin and S. M. Venkatesan. Approximation and intractability results for the maximum cut problem and its variants. *IEEE Transactions on Computers*, 40(1):110–113, 1991.

31 M. T. Hajiaghayi, G. Kortsarz, R. MacDavid, M. Purohit, and K. Sarpatwar. Approximation Algorithms for Connected Maximum Cut and Related Problems. In *ESA 2015*, pages 693–704, 2015.

32 T. Hanaka, H. L. Bodlaender, T. C. van der Zanden, and H. Ono. On the Maximum Weight Minimal Separator. In *TAMC 2017*, pages 304–318, 2017.

33 P. Hliněný and S. Oum. Finding Branch-Decompositions and Rank-Decompositions. *SIAM Journal on Computing*, 38(3):1012–1032, 2008.

34 R. M. Karp. *Reducibility among Combinatorial Problems*, pages 85–103. Springer US, Boston, MA, 1972.

35 K. Khoshkhah, M. K. Ghadikolaei, J. Monnot, and F. Sikora. Weighted Upper Edge Cover: Complexity and Approximability. In *WALCOM 2019*, pages 235–247, 2019.

36 M. Mahajan and V. Raman. Parameterizing above Guaranteed Values: MaxSat and MaxCut. *Journal of Algorithms*, 31(2):335–354, 1999.

37 G. I. Orlova and Y. G. Dorfman. Finding the maximal cut in a graph. *Engineering Cyvernetics*, 10(3):502–506, 1972.

38 S. Oum. Approximating Rank-width and Clique-width Quickly. *ACM Transactions on Algorithms*, 5(1):10:1–10:20, 2008.

39 S. Oum and P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006.

40 V. Raman and S. Saurabh. Improved fixed parameter tractable algorithms for two "edge" problems: MAXCUT and MAXDAG. *Information Processing Letters*, 104(2):65–72, 2007.

41 M. Rao. Clique-width of graphs defined by one-vertex extensions. *Discrete Mathematics*, 308(24):6157–6165, 2008.

42 N. Robertson and P. D. Seymour. Graph minors. V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

43 S. Saurabh and M. Zehavi. Parameterized Complexity of Multi-Node Hubs. In *IPEC 2018*, volume 115, pages 8:1–8:14, 2019.

44 S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *CVPR 2008*, pages 1–8, 2008.

45 M. Yannakakis and F. Gavril. Edge Dominating Sets in Graphs. *SIAM Journal on Applied Mathematics*, 38(3):364–372, 1980.

46 M. Zehavi. Maximum Minimal Vertex Cover Parameterized by Vertex Cover. *SIAM Journal on Discrete Mathematics*, 31(4):2440–2456, 2017.