Approximation Algorithms for Connected Maximum Cut and Related Problems

MohammadTaghi Hajiaghayi[1⋆], Guy Kortsarz[2⋆⋆], Robert MacDavid[2], Manish Purohit[1⋆⋆⋆], and Kanthi Sarpatwar[3†]

[1] University of Maryland, College Park, MD 20742, USA
`{hajiagha, manishp}@cs.umd.edu`
[2] Rutgers University - Camden, Camden, NJ 08102, USA
`guyk@camden.rutgers.edu, robertmacdavid@gmail.com`
[3] IBM T. J. Watson Research Center, Yorktown Heights, NY 10598
`sarpatwa@us.ibm.com`

Abstract. An instance of the Connected Maximum Cut problem consists of an undirected graph $G = (V, E)$ and the goal is to find a subset of vertices $S \subseteq V$ that maximizes the number of edges in the cut $\delta(S)$ such that the induced graph $G[S]$ is connected. We present the first non-trivial $\Omega(\frac{1}{\log n})$ approximation algorithm for the Connected Maximum Cut problem in general graphs using novel techniques. We then extend our algorithm to edge weighted case and obtain a poly-logarithmic approximation algorithm. Interestingly, in contrast to the classical Max-Cut problem that can be solved in polynomial time on planar graphs, we show that the Connected Maximum Cut problem remains NP-hard on unweighted, planar graphs. On the positive side, we obtain a polynomial time approximation scheme for the Connected Maximum Cut problem on planar graphs and more generally on bounded genus graphs.

1 Introduction

Submodular optimization problems have, in recent years, received a considerable amount of attention [1, 2, 3, 4, 7, 15, 27] in algorithmic research. In a general Submodular Maximization problem, we are given a non-negative submodular[4] function over the power set of a universe U of elements, $f : 2^U \rightarrow \mathbb{R}^+ \cup \{0\}$ and the goal is to find a subset $S \subseteq U$ that maximizes $f(S)$ so that S satisfies certain pre-specified constraints. In addition to their practical relevance, the study of submodular maximization problems has led to the development of several important theoretical techniques such as the continuous greedy method and multi-linear extensions [4] and the double greedy [2] algorithm, among others.

In this study, we are interested in the problem of maximizing a submodular set function over vertices of a graph, such that the selected vertices induce a connected subgraph. Motivated by applications in coverage over wireless networks, Kuo et al. [25] consider the

[4] A function f is called submodular if $f(S) + f(T) \geq f(S \cup T) + f(S \cap T)$ for all $S, T \subseteq U$.

problem of maximizing a monotone, submodular function f subject to connectivity and cardinality constraints of the form $|S| \leq k$ and provide an $\Omega(\frac{1}{\sqrt{k}})$ approximation algorithm. For a restricted class of monotone, submodular functions that includes the covering function[5], Khuller et al. [23] give a constant factor approximation to the problem of maximizing f subject to connectivity and cardinality constraints.

In the light of these results, it is rather surprising that no non-trivial approximation algorithms are known for the case of general (non-monotone) submodular functions. Formally, we are interested in the following problem, which we refer to as Connected Submodular Maximization (CSM): Given a simple, undirected graph $G = (V, E)$ and a non-negative submodular set function $f : 2^V \rightarrow \mathbb{R}^+ \cup \{0\}$, find a subset of vertices $S \subseteq V$ that maximizes $f(S)$ such that $G[S]$ is connected. We take the first but important step in this direction and study the problem in the case of one of the most important non-monotone submodular functions, namely the Cut function. Formally, given an undirected graph $G = (V, E)$, the goal is to find a subset $S \subseteq V$, such that $G[S]$ is connected and the number of edges that have exactly one end point in S, referred to as the cut function $\delta(S)$, is maximized. We refer to this as the Connected Maximum Cut problem. Further, we also consider an edge weighted variant of this problem, called the Weighted Connected Maximum Cut problem, where function to be maximized is the total weight of edges in the cut $\delta(S)$.

We now outline an application to the image segmentation problem that seeks to identify "objects" in an image. Graph based approaches for image segmentation [16, 28] represent each pixel as a vertex and weighted edges represent the dissimilarity (or similarity depending on the application) between adjacent pixels. Given such a graph, a connected set of pixels with a large weighted cut naturally corresponds to an object in the image. Vicente et al. [33] show that even for interactive image segmentation, techniques that require connectivity also perform significantly better that cut based methods alone.

Related Work

Max-Cut is a fundamental problem in combinatorial optimization that finds applications in diverse areas. A simple randomized algorithm that adds each vertex to S independently with probability $1/2$ gives a 0.5-approximate solution in expectation. In a breakthrough result, Goemans and Williamson [18] gave a 0.878-approximation algorithm using semidefinite programming and randomized rounding. Further, Khot et al. [22] showed that this factor is optimal assuming the Unique Games Conjecture. Interestingly, the Max-Cut problem can be optimally solved in polynomial time in planar graphs by a curious connection to the matching problem in the dual graph [20]. To the best of our knowledge, the Connected Maximum Cut problem has not been considered before our work. Haglin and Venkatesan [21] showed that a related problem, where we require both sides of the cut, namely S and $V \setminus S$, to be connected, is NP-hard in planar graphs.

We note that the well studied Maximum Leaf Spanning Tree (MLST) problem (e.g. see [31]) is a special case of the Connected Submodular Maximization problem. We also note that recent work on graph connectivity under vertex sampling leads to a simple constant approximation to the Connected Submodular Maximization for highly connected graphs, i.e., for graphs with $\Omega(\log n)$ vertex connectivity. Proofs of these claims are presented in the Appendix B and C respectively.

---

[5] In this context, a covering function is defined as $f(S) = \sum_{v \in \mathbb{N}^+(S)} weight(v)$ where $\mathbb{N}^+(S)$ is the closed neighborhood of the set of vertices S

We conclude this section by noting that connected variants of many classical combinatorial problems have been extensively studied in the literature and have been found to be useful. The best example for this is the Connected Dominating Set problem. Following the seminal work of Guha and Khuller [19], the problem has found extensive applications (with more than a thousand citations) in the domain of wireless ad hoc networks as a *virtual backbone* (e.g. see [9, 12]). Few other examples of connected variants of classic optimization problems include Group Steiner Tree [17] (which can be seen as a generalization of a connected variant of Set Cover), Connected Domatic Partition [5, 6], Connected Facility Location [13, 32], and Connected Vertex Cover [8].

Contribution and Techniques

Our key results can be summarized as follows.

1. We obtain the first $\Omega(\frac{1}{\log n})$ approximation algorithm for the Connected Maximum Cut (CMC) problem in general graphs. Often, for basic connectivity problems on graphs, one can obtain simple $O(\log n)$ approximation algorithms using a probabilistic embedding into trees with $O(\log n)$ stretch [14]. Similarly, using the cut-based decompositions given by Räcke [29], one can obtain $O(\log n)$ approximation algorithms for cut problems (e.g. Minimum Bisection). Interestingly, since the CMC problem has the flavors of both cut and connectivity problems simultaneously, neither of these approaches are applicable. Our novel approach is to look for α-thick trees, which are basically sub-trees with "high" degree sum on the leaves.

2. For the Weighted Connected Maximum Cut problem, we obtain an $\Omega(\frac{1}{\log^2 n})$ approximation algorithm. The basic idea is to group the edges into logarithmic number of weight classes and show that the problem on each weight class boils down to the special case where the weight of every edge is either 0 or 1.

3. We obtain a polynomial time approximation scheme for the CMC problem in planar graphs and more generally in bounded genus graphs. This requires the application of a stronger form of the edge contraction theorem by Demaine, Hajiaghayi and Kawarabayashi [11] that may be of independent interest.

4. We show that the CMC problem remains NP-hard even on unweighted, planar graphs. This is in stark contrast with the regular Max-Cut problem that can be solved optimally in planar graphs in polynomial time. We obtain a polynomial time reduction from a special case of 3-SAT called the Planar Monotone 3-SAT (PM-3SAT), to the CMC problem in planar graphs. This entails a delicate construction, exploiting the so called "rectilinear representation" of a PM-3SAT instance, to maintain planarity of the resulting CMC instance.

2 Approximation Algorithms for General Graphs

In this section, we consider the Connected Maximum Cut problem in general graphs. In fact, we provide an $\Omega(\frac{1}{\log n})$ approximation algorithm for the more general problem in which edges can have weight 0 or 1 and the objective is to maximize the number of edges of weight 1 in the cut. This generalization will be useful later in obtaining a poly-logarithmic approximation algorithm for arbitrary weighted graphs.

We denote the cut of a subset of vertices S in a graph G, i.e., the set of edges in G that are incident on exactly one vertex of S by $\delta_G(S)$ or when G is clear from context, just $\delta(S)$. Further, for two disjoint subsets of vertices S_1 and S_2 in G, we denote the set of edges

that have one end point in each of S_1 and S_2, by $\delta_G(S_1, S_2)$ or simply $\delta(S_1, S_2)$. The formal problem definition follows -

Problem Definition. {0,1}-Connected Maximum Cut (b-CMC): Given a graph $G = (V, E)$ and a weight function $w : E \to \{0, 1\}$, find a set $S \subset V$ that maximizes $\sum_{e \in \delta(S)} w(e)$ such that $G[S]$ induces a connected subgraph.

We call an edge of weight 0, a 0-edge and that of weight 1, a 1-edge. Further, let $w(\delta(S)) = \sum_{e \in \delta(S)} w(e)$ denote the weight of the cut, i.e., the number of 1-edges in the cut. We first start with a simple reduction rule that ensures that every vertex $v \in V$ has at least one 1-edge incident on it.

Claim 1 *Given a graph $G = (V, E)$, we can construct a graph $G' = (V', E')$ in polynomial time, such that every $v' \in V'$ has at least one 1-edge incident on it and G' has a b-CMC solution S' of weight at least ψ if and only if G has a b-CMC solution S of weight at least ψ.*

Proof. Let $v \in V$ be a vertex in G that has only 0-edges incident on it and let $\{v_1, v_2, \ldots, v_l\}$ denote the set of its neighbors. Consider the graph G' obtained from G by deleting v along with all its incident edges and adding 0-edges between every pair of its neighbors $\{v_i, v_j\}$ such that $\{v_i, v_j\} \notin E$. Let S denote a feasible solution of weight ψ in G. If $v \notin S$, then clearly $S' = S$ is the required solution in G'. If $v \in S$, we set $S' = S \setminus \{v\}$ and we claim that $G'[S']$ is connected if $G[S]$ is connected and $\sum_{e \in \delta_{G'}(S')} w(e) = \sum_{e \in \delta_G(S)} w(e)$. The latter part of the claim is true since all the edges that we delete and add are 0-edges. To prove the former part, notice that if v is not a cut vertex in $G[S]$ then $G[S']$ must be connected. On the other hand, even if v is a cut vertex, the new edges added among all pairs of v's neighbors ensure that $G[S']$ is connected. Finally, to prove the other direction, suppose we have a feasible solution S' of weight ψ in G'. Now, if $G[S']$ is connected, then $S = S'$ is a feasible solution in G of weight ψ. Otherwise, set $S = S' \cup \{v\}$. Since v creates a path between all pairs of its neighbors, $G[S]$ is connected if $G'[S']$ is connected and is thus a feasible solution of the same weight. The proof of the lemma follows from induction. $\square$

From now on, we will assume, without loss of generality, that every vertex of G has at least one 1-edge incident on it. We now introduce some new definitions that would help us to present the main algorithmic ideas. We denote by $W_G(v)$ the total weight of edges incident on a vertex v in G, i.e., $W_G(v) = \sum_{e : v \in e} w(e)$. In other words, $W_G(v)$ is total number of 1-edges incident on v. Further let η be the total number of 1-edges in the graph. The following notion of an α-thick tree is a crucial component of our algorithm.

Definition 1 (α-Thick Tree). *Let $G = (V, E)$ be a graph with n vertices and η 1-edges. A subtree $T \subseteq G$ (not necessarily spanning), with leaf set L, is said to be α-thick if $\sum_{v \in L} W_G(v) \geq \alpha \eta$.*

The following lemma shows that this notion of an α-thick tree is intimately connected with the b-CMC problem.

Lemma 1 *For any $\alpha > 0$, given a polynomial time algorithm A that computes an α-thick tree T of a graph G, we can obtain an $\frac{\alpha}{4}$-approximation algorithm for the b-CMC problem on G.*

Proof. Given a graph $G = (V, E)$ and weight function $w : E \to \{0, 1\}$, we use Algorithm A to compute an α-thick tree T, with leaf set L. Let m_L denote the number of 1-edges in $G[L]$, the subgraph induced by L in the graph G. We now partition L into two disjoint sets L_1 and L_2 such that the number of 1-edges in $\delta(L_1, L_2) \geq \frac{m_L}{2}$. This can be done by applying the standard randomized algorithm for Max-Cut (e.g. see [26]) on $G[L]$ after deleting all the 0-edges. Now, consider the two connected subgraphs $T \setminus L_1$ and $T \setminus L_2$. We first claim that every 1-edge in $\delta(L)$ belongs to either $\delta(T \setminus L_1)$ or $\delta(T \setminus L_2)$. Indeed, any 1-edge e in $\delta(L)$, belongs to one of the four possible sets, namely $\delta(L_2, T \setminus L)$, $\delta(L_1, V \setminus T)$, $\delta(L_1, T \setminus L)$ and $\delta(L_2, V \setminus T)$. In the first two cases, e belongs to $\delta(T \setminus L_2)$ while in the last two cases, e belongs $\delta(T \setminus L_1)$, hence the claim. Further, every 1-edge in $\delta(L_1, L_2)$ belongs to both $\delta(T \setminus L_1)$ and $\delta(T \setminus L_2)$. Hence, we have -

$$\sum_{e \in \delta(T \setminus L_1)} w(e) + \sum_{e \in \delta(T \setminus L_2)} w(e) = \sum_{e \in \delta(L)} w(e) + 2 \sum_{e \in \delta(L_1, L_2)} w(e) \tag{1}$$

$$\geq \sum_{e \in \delta(L)} w(e) + m_L \geq \frac{1}{2} \sum_{v \in L} W_G(v) \geq \frac{\alpha \eta}{2} \tag{2}$$

Hence, the better of the two solutions $T \setminus L_1$ or $T \setminus L_2$ is guaranteed to have a cut of weight at least $\frac{\alpha \eta}{4}$, where η is the total number of 1-edges in G. To complete the proof we note that for any optimal solution OPT, $w(\delta(OPT)) \leq \eta$. $\square$

Thus, if we have an algorithm to compute α-thick trees, Lemma 1 provides an $\Omega(\alpha)$-approximation algorithm for the b-CMC problem. Unfortunately, there exist graphs that do not contain α-thick trees for any non-trivial value of α. For example, let G be a *path graph* with n vertices and $m = n - 1$ 1-edges. It is easy to see that for any subtree T, the sum of degrees of the leaves is at most 4. In spite of this setback, we show that the notion of α-thick trees is still useful in obtaining a good approximation algorithm for the b-CMC problem. In particular, Lemma 3 and Theorem 1 show that path graph is the *only* bad case, i.e., if the graph G does not have a long induced path, then one can find an $\Omega(\frac{1}{\log n})$-thick tree. Lemma 2 shows that we can assume without loss of generality that the b-CMC instance does not have such a long induced path.

Shrinking Thin Paths. A natural idea to handle the above "bad" case is to get rid of such long paths that contain only vertices of degree two by contracting the edges. We refer to a path that only contains vertices of degree two as a d-2 path. Further, we define the length of a d-2 path as the number of *vertices* (of degree two) that it contains. The following lemma shows that we can assume without loss of generality that the graph G contains no "long" d-2 paths.

Lemma 2 *Given a graph G, we can construct, in polynomial time, a graph G' with no d-2 paths of length ≥ 3 such that G' has a b-CMC solution S' of cut weight $(w(\delta(S')))$ at least ψ if and only if G has a b-CMC solution S of cut weight at least ψ. Further, given the solution S' of G', we can recover S in polynomial time.*

Proof. We may assume that G is connected, because otherwise we can handle each component separately. We further assume that G is not a simple cycle, otherwise it is trivial to solve such an instance. If G does not have a d-2 path of length ≥ 3, then trivially we

have $G' = G$. Otherwise, let $\wp = [v_0, e_0, v_1, e_1, v_2, e_2, v_3]$ be a path in G such that v_1, v_2 and v_3 have degree two and $deg(v_0) \neq 2$. Note that such a path $\wp$ must exist as G is not a simple cycle. We now perform the following operation on G to obtain a new graph G_{new}: Delete these elements $\{e_0, v_1, e_1, v_2, e_2\}$. Add a new vertex v_{new} and edges $e'_0 = (v_0, v_{new})$ and $e'_1 = (v_{new}, v_3)$. Since $deg(v_0) \neq 2$ and $deg(v_3) = 2$, we are guaranteed that $v_0 \neq v_3$ and hence we do not introduce any multi-edges. The weights on the new edges are determined as follows - Let $n_\wp$ denote the number of 1-edges in $E_\wp = \{e_0, e_1, e_2\}$. If $n_\wp \geq 2$, we set $w(e'_0) = w(e'_1) = 1$. If $n_\wp = 1$, then we set $w(e'_0) = 0$ and $w(e'_1) = 1$. Otherwise, we set $w(e'_0) = w(e'_1) = 0$. We claim that G_{new} has a b-CMC solution S' of cut weight at least ψ if and only if G has a solution S of cut weight at least ψ.

Let us first assume that there is a set S in G that is a solution to the b-CMC problem with cut weight ψ. We now show that there exists a S' in G_{new} that is a solution to the b-CMC problem with cut weight at least ψ. The proof in this direction is done for three possible cases, based on the cardinality of $\delta(S) \cap E_\wp$. We note that $|\delta(S) \cap E_\wp|$ is ≤ 2, since $G[S]$ must be connected.

Case 1. $|\delta_G(S) \cap E_\wp| = 2$. Note that since S is connected, we must have either (i) $S \subseteq \{v_1, v_2\}$ or (ii) $\{v_0, v_3\} \subseteq S$. In the former case, we set $S' = \{v_{new}\}$ and the claim follows by the definition of $w(e'_0)$ and $w(e'_1)$. In the latter case, we set $S' = S \setminus \{v_1, v_2\}$. Since v_1 and v_2 are vertices of degree two, $G_{new}[S']$ is connected. Further, every edge $e \in \delta_G(S) \setminus E_\wp$ also belongs to $\delta_{G_{new}}(S')$. The claim follows once we observe that both e'_0 and e'_1 are in $\delta_{G_{new}}(S')$.

Case 2. $|\delta_G(S) \cap E_\wp| = 1$. In this case, we must have either $v_0 \in S$ or $v_3 \in S$ but not both. Let us first assume $v_0 \in S$. We set $S' = (S \cup \{v_{new}\}) \setminus \{v_1, v_2\}$. It is clear that if $G[S]$ is connected, so is $G_{new}[S']$. Due to the removal of v_1 and v_2, we have $\delta_G(S) \setminus \delta_{G_{new}}(S') = \{e_i\}$ for some edge $e_i \in E_\wp$. On the other hand, due to the addition of v_{new}, we have $\delta_{G_{new}}(S') \setminus \delta_G(S) = \{e'_1\}$ and the claim follows since $w(e'_1) \geq w(e_i)$ for any $e_i \in E_\wp$. Now assume that $v_3 \in S$. In this case, we set $S' = S \setminus \{v_1, v_2\}$. Since $v_{new} \notin S'$, we again have $e'_1 \in \delta_{G_{new}}(S')$ and the proof follows as above.

Case 3. $|\delta_G(S) \cap E_\wp| = 0$. In this case, one of the following holds, either (i) $\{v_0, v_1, v_2, v_3\} \subseteq S$ or (ii) $\{v_0, v_1, v_2, v_3\} \cap S = \phi$. If the latter is true, the proof is trivial by setting $S' = S$. In the former case, we set $S' = S \setminus \{v_1, v_2\} \cup \{v_{new}\}$. The addition of v_{new} maintains connectivity between v_0 and v_3 and hence since S is connected, so is S'. Further, we have $\delta_G(S) = \delta_{G_{new}}(S')$ since no edge in $\delta_G(S)$ in incident on v_1 or v_2.

In order to prove the other direction, we assume that S' is a solution to the b-CMC problem on G_{new} with a cut weight of ψ. We now construct a set S that is a solution to b-CMC on G of weight at least ψ. The proof proceeds in three cases similarly.

Case 1. Both $e'_0 \in \delta_{G_{new}}(S')$ and $e'_1 \in \delta_{G_{new}}(S')$. One of the following holds - (i) $S' = \{v_{new}\}$ or (ii) $\{v_0, v_3\} \subseteq S'$. In the former case, let S be the subset of $\{v_1, v_2\}$ having the largest weight cut. By construction, we have that weight of the cut $\delta(S)$ is at least the sum of weights of e'_0 and e'_1. For the latter, let S to be the best among $S', S' \cup \{v_1\}$, and $S' \cup \{v_2\}$ and the proof follows as above.

Case 2. Either $e'_0 \in \delta_{G_{new}}(S')$ or $e'_1 \in \delta_{G_{new}}(S')$ but not both. Let e_{max} be the edge of maximum weight in $E_\wp$. The edge e_{max} splits the path $\wp$ into two connected components one containing v_0, call it $\wp_0$ and the other containing v_3, call it $\wp_3$. Now to construct S, we delete v_{new} from S' (if it contains it) and add the component $\wp_0$ if $v_0 \in S'$ or the component $\wp_3$ if $v_3 \in S'$. Again connectivity is clearly preserved. We now argue that the cut weight is also preserved. Indeed, this is true since we have that $w(e_{max}) \geq max(w_{e'_0}, w_{e'_1})$ and the rest of the cut edges in S' remain as they are in S.

Case 3. None of e'_0, e'_1 belong to $\delta_{G_{new}}(S')$. In this case, if $v_{new} \notin S'$, then trivially $S = S'$ works. Otherwise, we set $S = S' \cup \{v_1, v_2\}$. It is easy to observe that both connectivity and all the cut edges are preserved in this case.

Now, to construct G', we repeatedly apply the above contraction as long as possible. This will clearly take polynomial time as in each iteration, we reduce the number of degree-2 vertices by 1. Hence we have the claim. □

Spanning Tree with Many Leaves. Assuming that the graph has no long d-2 paths, the following lemma shows that we can find a spanning tree T that has $\Omega(n)$ leaves. Note that Claim 1 now guarantees that there are $\Omega(n)$ 1-edges incident on the leaves of T.

Lemma 3 *Given a graph $G = (V, E)$ with no d-2 paths of length ≥ 3, we can obtain, in polynomial time, a spanning tree $T = (V, E_T)$ with at least $\frac{n}{14}$ leaves.*

Proof. Let T be any spanning tree of G. We note that although G does not have d-2 paths of length ≥ 3, such a guarantee does not hold for paths in T. Suppose that there is a d-2 path $\wp$ of length 7 in T. Let the vertices of this path be numbered $v_1, v_2, \ldots, v_7$ and consider the vertices v_3, v_4, v_5. Since G does not have any d-2 path of length 3, there is a vertex $v_i, i \in \{3, 4, 5\}$ such that $deg_G(v_i) \geq 3$. We now add an edge $e = \{v_i, w\}$ in $G \setminus T$ to the tree T. The cycle C that is created as a result must contain either the edge $\{v_1, v_2\}$ or the edge $\{v_6, v_7\}$. We delete this edge to obtain a new spanning tree T'. It is easy to observe that the number of vertices of degree two in T' is strictly less than that in T. This is because, although the new edge $\{v_i, w\}$ can cause w to have degree two in T', we are guaranteed that the vertex v_i will have degree three and vertices v_1 and v_2 (or v_6 and v_7) will have degree one. Hence, as long as there are d-2 paths of length 7 in T, the number of vertices of degree two can be strictly decreased. Thus this process must terminate in at most n steps and the final tree $T^{(1)}$ obtained does not have any d-2 paths of length ≥ 7.

We now show that the tree $T^{(1)}$ contains $\Omega(n)$ leaves by a simple charging argument. Let the tree $T^{(1)}$ be rooted at an arbitrary vertex. We assign each vertex of $T^{(1)}$ a token and redistribute them in the following way : Every vertex v of degree two in $T^{(1)}$ gives its token to its first non degree two descendant, breaking ties arbitrarily. Since there is no d-2 path of length ≥ 7, each non degree two vertex collects at most 7 tokens. Hence, the number of vertices not having degree two in $T^{(1)}$ is at least $\frac{n}{7}$. Further, since the average degree of all vertices in a tree is at most 2, a simple averaging argument shows that $T^{(1)}$ must contain at least $\frac{n}{14}$ vertices of degree one, i.e., $\frac{n}{14}$ leaves. □

Obtaining an $\Omega(\frac{1}{\log n})$ Approximation

We now have all the ingredients required to obtain the $\Omega(\frac{1}{\log n})$ approximation algorithm. We observe that if the graph G is sparse, i.e. $\eta \leq cn \log n$ (for a suitable constant c), then the tree obtained by using Lemma 3 is an $\Omega(\frac{1}{\log n})$-thick tree and thus we obtain the required approximate solution in this case. On the other hand, if the graph G is sparse, then we use Lemma 3 to obtain a spanning tree, delete the leaves of this tree, and then repeat this procedure until we have no more vertices left. Since, we delete a constant fraction of vertices in each iteration, the total number of iterations is $O(\log n)$. We then choose the "best" tree out of the $O(\log n)$ trees so obtained and show that it must be an α-thick tree, with $\alpha = \Omega(\frac{1}{\log n})$. Finally, using Lemma 1, we obtain an $\Omega(\frac{1}{\log n})$ approximate solution as desired. We refer to Algorithm 1 for the detailed algorithm.

```
 1  Input: Graph $G = (V, E)$
 2  Output: A subset $S \subseteq V$, such that $G[S]$ is connected
 3  Set $G_1(V_1, E_1) = G$, $n_1 = |V_1|$
 4  Let $\eta \leftarrow$ Number of 1-edges in $G$
 5  Use Lemma 3 to obtain a spanning tree $T_1$ of $G_1$ with leaf set $L_1$
 6  if $\eta \leq cn \log n$ then
 7  │    Use Lemma 1 on $T_1$ to obtain a set connected $S$
 8  │    return $S$
 9  end
10  $i = 1$
11  while $G_i \neq \phi$ do
12  │    $E_{i+1} \leftarrow E_i \setminus (E[L_i] \cup \delta(L_i))$
13  │    $V_{i+1} \leftarrow V_i \setminus L_i$, $n_{i+1} = |V_{i+1}|$
14  │    Contract degree-2 vertices in $G_{i+1}$
15  │    Use Lemma 3 to obtain a spanning tree $T_{i+1}$ of $G_{i+1}$ with leaf set $L_{i+1}$
16  │    $i = i + 1$
17  end
18  Choose $j = \arg\max_i(\sum_{v \in L_i} deg_G(v))$
19  Use Lemma 1 on $T_j$ to obtain a connected set $S$
20  return $S$
```

Algorithm 1: Finding α-thick trees

Theorem 1. *Algorithm 1 gives an $\Omega(\frac{1}{\log n})$ approximate solution for the b-CMC problem.*

Proof. Let us assume that $\eta \leq cn \log n$ (for some constant c). Now, Lemma 3 and Claim 1 together imply that $\sum_{v \in L_1} W_G(v) = \Omega(n)$. Further, since we have $w(\delta(OPT)) \leq \eta \leq cn \log n$, T is an α-thick tree for some $\alpha = \Omega(\frac{1}{\log n})$. Hence, we obtain an $\Omega(\frac{1}{\log n})$ approximate solution using Lemma 1.

On the other hand, if $\eta > cn \log n$, we show that at least one of the trees T_i obtained by the repeated applications of the Lemma 3 is an α-thick tree T of G for $\alpha = \Omega(\frac{1}{\log n})$. We first observe that the While loop in Step 11 runs for at most $O(\log n)$ iterations. This is because we delete $\Omega(n_i)$ leaves in each iteration and hence after $k = O(\log n)$ iterations, we get $G_k = \phi$. We now count the number of 1-edges "lost" in each iteration. We recall that $W_G(v)$ is the total number of 1-edges incident on v in a graph G. In an iteration i, the number of 1-edges lost at Step 12 is at most $\sum_{v \in L_i} W_{G_i}(v)$. In addition, we may lose a total of at most $2n \leq \frac{2\eta}{c \log n}$ edges due to the contraction of degree two vertices in Step 14. Suppose for the sake of contradiction that $\sum_{v \in L_i} W_G(v) < \frac{\eta}{d \log n}, \forall 1 \leq i \leq k$ where d is a suitable constant. Then the total number of 1-edges lost in $k = O(\log n)$ iterations is at most

$$\sum_{i=1}^{k}(\sum_{v \in L_i} W_{G_i}(v)) + \frac{2\eta}{c \log n} < \sum_{i=1}^{k} \frac{\eta}{d \log n} + \frac{2\eta}{c \log n} = \frac{\eta}{\hat{d}} + \frac{\eta}{c \log n} < \eta$$

The equality follows for a suitable constant $\hat{d}$ as $k = O(\log n)$. The final inequality holds for a suitable choice of the constants c and d. But this is a contradiction since we have $G_k = \phi$.

Since we choose j to be the best iteration, we have $\sum_{v \in L_j} W_G(v) \geq \frac{\eta}{d \log n}$ for some constant d. Hence the tree T_j is an α-thick tree of G for $\alpha = \frac{1}{d \log n}$ and the theorem follows by Lemma 1. □

General Weighted Graphs

We now consider the Weighted Connected Maximum Cut (WCMC) problem. Formally, we are given a graph $G = (V, E)$ and a weight function $w : E \to \mathbb{R}^+ \cup \{0\}$. The goal is to find a subset S of vertices that induces a connected subgraph and maximizes the quantity $\sum_{e \in \delta(S)} w(e)$. We obtain a $\Omega(\frac{1}{\log^2 n})$ approximation algorithm for this problem. Our basic strategy is to group edges having nearly the same weight into a class and thus create $O(\log n)$ classes. We then solve the b-CMC problem for each class independently and return the best solution.

---

1 Input: Connected graph $G = (V, E)$ with $|V| = n$ and $|E| = m$; Weight function, $w : E \to \mathbb{R}^+ \cup 0$; $\epsilon > 0$;
2 Output: A subset $S \subseteq V$, such that $G[S]$ is connected;
3 Let w_{max} be the maximum weight over any edge of the graph;
4 Define, $w_0 = \frac{\epsilon w_{max}}{m}$ and $w_i = w_0(1 + \epsilon)^i$, for $i \in [\log_{1+\epsilon} \frac{m}{\epsilon}]$;
5 for $i \in [0, \log_{1+\epsilon} \frac{m}{\epsilon}]$ **do**
6   **for** $e \in E$ **do**
7   **if** $w_i \leq w(e) < w_{i+1}$ **then**
8 | $w_i'(e) = 1$;
9   **end**
10   **else**
11 | $w_i'(e) = 0$;
12   **end**
13   Using Theorem 1, solve for the connected subset S_i;
14   **end**
15 end
16 return S_{best}, such that $best = \underset{i \in [0, \log \frac{n}{\epsilon}]}{\arg\max} \sum_{e \in \delta(S_i)} w(e)$;

Algorithm 2: Algorithm for the Weighted Connected Maximum Cut problem.

Theorem 1. *Algorithm 2 gives a $\Omega(\frac{1}{\log^2 n})$ approximation guarantee for the Weighted Connected Maximum Cut problem.*

Proof. Let OPT be an optimal solution for a given instance of the problem and let $\psi = \sum_{e \in \delta(OPT)} w(e)$. Also, let $\epsilon \in (0, 1]$. Since we have that $\psi \geq w_{max}$, we can reset the weights of those edges with weight $< \frac{\epsilon w_{max}}{m}$ to 0 and assume that $w_{min} \geq \frac{\epsilon w_{max}}{m}$ where w_{min} denotes the weight of the minimum (non zero) weight edge. Let E_i be the set of edges e such that $w_i \leq w(e) < w_{i+1}$ and finally let $OPT_i = \delta(OPT) \cap E_i$. We now claim that $\sum_{e \in OPT_i} w(e) = O((1 + \epsilon) \log n \sum_{e \in \delta(S_i)} w(e))$. This immediately gives us that $\sum_{e \in \delta(S_{best})} w(e) = \Omega(\frac{\sum_{e \in OPT} w(e)}{(1+\epsilon) \log n \log_{1+\epsilon} \frac{m}{\epsilon}}) = \Omega(\frac{1}{\log^2 n})(\sum_{e \in OPT} w(e))$.

We now prove the claim. Consider solving the b-CMC instance with weight function w_i'. Clearly OPT is a feasible solution to this instance and we have $\sum_{e \in \delta(OPT)} w_i'(e) = \sum_{e \in OPT_i} w_i'(e) \leq O(\log n \sum_{e \in \delta(S_i)} w_i'(e))$. The previous inequality holds as S_i is guaranteed to be an $\Omega(\frac{1}{\log n})$-approximate solution by Theorem 1. Now, we have $\sum_{e \in OPT_i} w(e) \leq (1+\epsilon)w_i \sum_{e \in OPT_i} w_i'(e) \leq O((1+\epsilon)w_i \log n \sum_{e \in \delta(S_i)} w_i'(e)) \leq O((1+\epsilon) \log n \sum_{e \in \delta(S_i)} w(e))$. Hence, the claim. $\square$

3 CMC in Planar and Bounded Genus Graphs

In this section, we consider the CMC problem in planar graphs and more generally, in graphs with genus bounded by a constant. We show that the CMC problem has a PTAS in bounded genus graphs.

PTAS for Bounded Genus Graphs.

We use the following (paraphrased) contraction decomposition theorem by Demaine, Haji-aghayi and Kawarabayashi [11].

Theorem 2. ([11]) *For a bounded-genus graph G and an integer k, the edges of G can be partitioned into k color classes such that contracting all the edges in any color class leads to a graph with treewidth $O(k)$. Further, the color classes are obtained by a radial coloring and have the following property: If edge $e = (u, v)$ is in class i, then every edge e' such that $e' \cap e \neq \phi$ is in class $i - 1$ or i or $i + 1$.*

Given a graph G of constant genus, we use Theorem 2 appropriately to obtain a graph H with constant treewidth. In Appendix A, we show that one can solve the CMC problem optimally in polynomial time on graphs with constant treewidth.

Theorem 3. *If the CMC problem can be solved optimally on graphs of constant treewidth, then there exists a polynomial time $(1 - \epsilon)$ approximation algorithm for the CMC problem on bounded genus graphs (and hence on planar graphs).*

Proof. Let $G = (V, E)$ be the graph of genus bounded by a constant and let S denote the optimal CMC of G and $\psi = |\delta(S)|$ be its size. Using Theorem 2 with $k = \frac{3}{\epsilon}$, we obtain a partition of the edges E into $\frac{3}{\epsilon}$ color classes namely $C_1, C_2, \ldots, C_{\frac{3}{\epsilon}}$. We further group three consecutive color classes into $\frac{1}{\epsilon}$ groups $G_1, \ldots, G_{\frac{1}{\epsilon}}$ where $G_j = C_{3j-2} \cup C_{3j-1} \cup C_{3j}$. Let G_{j^*} denote the group that intersects the least with the optimal connected max cut of G, i.e., $j^* = \arg \min_j (|G_j \cap \delta(S)|)$[6]. As the $\frac{1}{\epsilon}$ groups partition the edges, we have $|G_{j^*} \cap \delta(S)| \leq \epsilon \psi$. Let $i = 3j^* - 1$, so that $G_{j^*} = C_{i-1} \cup C_i \cup C_{i+1}$. Let $H = (V_H, E_H)$ denote the graph of treewidth $O(\frac{1}{\epsilon})$ obtained by contracting all edges of color C_i.

We first show that H has a CMC of size at least $(1 - \epsilon)\psi$. For a vertex $v \in V_H$, let $\mu(v) \subseteq V$ denote the set of vertices of G that have merged together to form v due to the contraction. We define a subset $S' \subset V_H$ as $S' = \{v \in V_H \mid \mu(v) \cap S \neq \phi\}$. Note that because we contract edges (and not delete them), S' remains connected. We claim that $|\delta(S')| \geq (1 - \epsilon)\psi$. Let $e = (u, v)$ be an edge in $\delta(S)$. Now $e \notin \delta(S')$ implies that at least one edge e' such that $e' \cap e \neq \phi$ has been contracted. By the property guaranteed by Theorem 2, we have that $e \in G_{j^*}$. Hence we have, $|\delta(S')| \geq |\delta(S) \setminus G_{j^*}| = |\delta(S)| - |G_{j^*} \cap \delta(S)| \geq (1 - \epsilon)\psi$.

Finally, given a connected max cut of size ψ in H, we can recover a connected max cut of size at least ψ in G by simply un-contracting all the contracted edges. Hence, by solving the CMC problem on H optimally, we obtain a $(1 - \epsilon)$ approximate solution in G. $\qquad \square$

---

[6] We "guess" j^* by trying out all the $\frac{1}{\epsilon}$ possibilities

NP-hardness in planar graphs

We now describe a non-trivial polynomial time reduction of a 3-SAT variant known as Planar Monotone 3-SAT (PM-3SAT) to the CMC problem on a planar graph, thereby proving that the latter is NP-hard. The following reduction is interesting as the classical Max-Cut problem can be solved optimally in polynomial time on planar graphs using duality. In fact, it was earlier claimed that even CMC can be solved similarly [21].

An instance of PM-3SAT is a 3-CNF boolean formula ϕ such that -
a) A clause contains either all positive literals or all negative literals.
b) The associated bipartite graph G_ϕ[7] is planar.
c) Furthermore, G_ϕ has monotone, rectilinear representation. We refer the reader to Berg and Khosravi [10] for a complete description. Figure 1a illustrates the rectilinear representation by a simple example.

(a) Monotone Rectilinear Representation

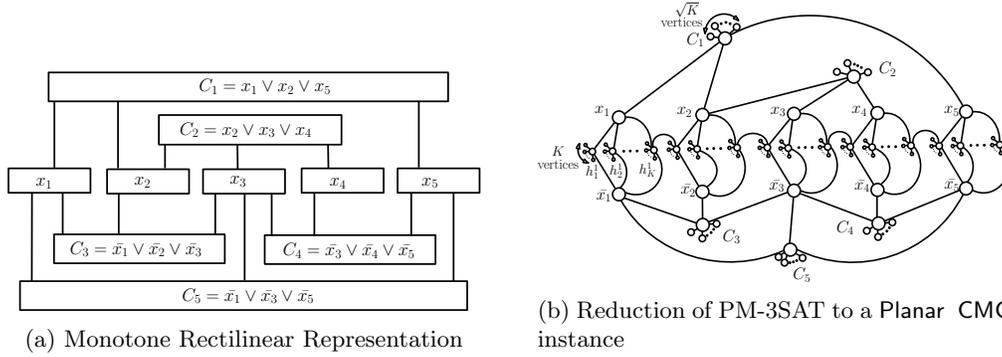(b) Reduction of PM-3SAT to a Planar CMC instance

Fig. 1: Example illustrating the rectilinear representation and the reduction to a Planar CMC instance of the formula $(x_1 \vee x_2 \vee x_5) \wedge (x_2 \vee x_3 \vee x_4) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_3 \vee \bar{x}_4 \vee \bar{x}_5) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_5)$.

Given such an instance, the PM-3SAT problem is to decide whether the boolean formula is satisfiable or not. Berg and Khosravi [10] show that the PM-3SAT problem is NP-complete.

The Reduction. Given a PM-3SAT formula ϕ, with a rectilinear representation, we obtain a polynomial time reduction to a Planar CMC instance, there by showing that the latter is NP-hard. Let $\{x_i\}_{i=1}^n$ denote the variables of the PM-3SAT instance and $\{C_j\}_{j=1}^m$ denote the clauses. We construct a planar graph H_ϕ as follows. For every variable x_i, we construct the following gadget: We create two vertices $v(x_i)$ and $v(\bar{x}_i)$ corresponding to the literals x_i and $\bar{x}_i$. Additionally, we have $K > m^2$ "helper" vertices, $h_1^i, h_2^i, \ldots, h_K^i$ such that each h_k^i is adjacent to both x_i and $\bar{x}_i$. Further, for every h_k^i we add a set L_k^i of K new vertices that are adjacent only to h_k^i. Now, in the rectilinear representation of the PM-3SAT, we replace each variable rectangle by the above gadget. For two adjacent variable rectangles in the rectilinear representation, say x_i and x_{i+1}, we connect the helpers h_K^i and h_1^{i+1}. For every clause C_j, H_ϕ has a corresponding vertex $v(C_j)$ with edges to the three literals in the clause. Finally, for each vertex $v(C_j)$, we add a set L_j of $\sqrt{K}$ new vertices adjacent

---

[7] G_ϕ has a vertex for each clause and each variable and an edge between a clause and the variables that it contains

only to $v(C_j)$. It is easy to observe that the reduction maintains the planarity of the graph. Figure 1b illustrates the reduction by an example.

We show the following theorem that proves the Planar Connected Maximum Cut problem is NP-hard.

Theorem 4. *Let H_ϕ denote an instance of the planar CMC problem corresponding to an instance ϕ of PM-3SAT obtained as per the reduction above. Then, the formula ϕ is satisfiable if and only if there is a solution S to the CMC problem on H_ϕ with $|\delta_{H_\phi}(S)| \geq m\sqrt{K} + nK + nK^2$.*

Proof. For brevity, we denote $\delta_{H_\phi}(S)$ as $\delta(S)$ in the rest of the proof.

Forward direction. Assume that ϕ is satisfiable under an assignment A. We now show that we can construct a set S with the required properties. Let $\{l_i\}_{i \in [n]}$ be the set of literals that are true in A. We define $S = \{v(l_i)\}_{i \in [n]} \cup \{C_j\}_{j \in [m]} \cup \{h_k^i\}_{i \in [n], k \in [K]}$, i.e., the set of vertices corresponding to the true literals, all the clauses and all the helper vertices. By construction, the set of all helper vertices and one literal of each variable induces a connected subgraph. Further, since in a satisfying assignment every clause has at least one true literal, the constructed set S is connected. We now show that $|\delta(S)| \geq m\sqrt{K} + nK + nK^2$. Indeed, $\delta(S)$ contains all the edges corresponding to the one degree vertices incident on clauses and all the helpers. This contributes a profit of $m\sqrt{K} + nK^2$. Also, since no vertex corresponding to a false literal is included in S but all helpers are in S, we get an additional profit of K for each variable. Hence, we have the claim.

Reverse direction. Assume that S is a subset of vertices in H_ϕ such that $H_\phi[S]$ is connected and $|\delta(S)| \geq m\sqrt{K} + nK + nK^2$. We now show that ϕ is satisfiable. We may assume that S is an optimal solution (since optimal solution will satisfy these properties, if a sub-optimal solution does). We first observe that at least one of the (two) literals for each variable must be chosen into S. Indeed, if this is not the case for some variable, for $H_\phi[S]$ to be remain connected, none of the helper vertices corresponding to that variable can be chosen. This implies that the maximum possible value for $|\delta(S)| \leq (n-1)K^2 + m\sqrt{K} + 3m + 2(n-1)K$ (this is the number of remaining edges)$< nK^2$ (since $K > m^2$) $< m\sqrt{K} + nK + nK^2$, a contradiction. We now show that every helper vertex must be included in S. Assume that this is not true and let h_k^i be some helper vertex not added to S. We note that none of the K degree one vertices in L_k^i can be in S because $H_\phi[S]$ must be connected. Now, consider the solution S' formed by adding h_k^i to S. Since at least one vertices $v(x_i)$ or $v(\bar{x}_i)$ is in S, if $H_\phi[S]$ is connected, so is $H_\phi[S']$. Further, the total number of edges in the cut increases by $K - 2$. This is a contradiction to the fact that S is an optimal solution. Hence, every helper vertex h_k^i belongs to the solution S. We now show that, no two literals of the same variable are chosen into S. Assume the contrary and let $v(x_i)$, $v(\bar{x}_i)$ both be chosen into S. We claim that removing one of these two literals will strictly improve the solution. Indeed, consider removing $v(x_i)$ from S. Clearly, we gain all the edges from $v(x_i)$ to all the helper vertices corresponding to this variable. Thus we gain at least K edges. We now bound the loss incurred. In the worst case, removing $v(x_i)$ from S might force the removal of all the clause vertices due to the connectivity restriction. But this would lead to a loss of at most $m\sqrt{K} + 3m < K$. Hence, we arrive at a contradiction that S is an optimal solution. Therefore, exactly one literal vertex corresponding to each variable is included in S. Finally, we observe that all the clauses must be included in S. Assume this is not true and that $m' < m$ clause vertices are in S. Now the total cut is $nK + nK^2 + m'\sqrt{K} < nK + nK^2 + m\sqrt{K}$, which is again a contradiction. Now, the optimal solution S gives a natural assignment to the PM-3SAT instance: a literal is set to TRUE if its corresponding vertex is included in S. Since,

every clause vertex belongs to S, which in turn is connected, it must contain a TRUE literal and hence the assignment satisfies ϕ. □

References

[1] Ashwinkumar Badanidiyuru and Jan Vondrák. Fast algorithms for maximizing submodular functions. In *SODA*, pages 1497–1514. SIAM, 2014.

[2] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. In *FOCS*, pages 649–658. IEEE, 2012.

[3] Niv Buchbinder, Moran Feldman, Joseph Naor, and Roy Schwartz. Submodular maximization with cardinality constraints. In *SODA*, pages 1433–1452. SIAM, 2014.

[4] Gruia Calinescu, Chandra Chekuri, Martin Pál, and Jan Vondrák. Maximizing a submodular set function subject to a matroid constraint. In *IPCO*, pages 182–196. Springer, 2007.

[5] Keren Censor-Hillel, Mohsen Ghaffari, George Giakkoupis, Bernhard Haeupler, and Fabian Kuhn. Tight bounds on vertex connectivity under vertex sampling. In *SODA*. SIAM, 2015.

[6] Keren Censor-Hillel, Mohsen Ghaffari, and Fabian Kuhn. A new perspective on vertex connectivity. In *SODA*, pages 546–561. SIAM, 2014.

[7] Chandra Chekuri and Alina Ene. Submodular cost allocation problem and applications. In *ICALP*, pages 354–366. Springer, 2011.

[8] Marek Cygan. Deterministic parameterized connected vertex cover. In *SWAT*, pages 95–106. Springer, 2012.

[9] Bevan Das and Vaduvur Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. In *ICC*, volume 1, pages 376–380. IEEE, 1997.

[10] Mark de Berg and Amirali Khosravi. Finding perfect auto-partitions is np-hard. In *EuroCG 2008*, pages 255–258. Citeseer, 2008.

[11] Erik D Demaine, MohammadTaghi Hajiaghayi, and Ken-ichi Kawarabayashi. Contraction decomposition in H-minor-free graphs and algorithmic applications. In *STOC*, pages 441–450. ACM, 2011.

[12] D.Z. Du and P.J. Wan. *Connected dominating set: theory and applications*. Springer optimization and its applications. Springer New York, 2013.

[13] Friedrich Eisenbrand, Fabrizio Grandoni, Thomas Rothvoß, and Guido Schäfer. Approximating connected facility location problems via random facility sampling and core detouring. In *SODA*, pages 1174–1183, 2008.

[14] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *STOC*, pages 448–455. ACM, 2003.

[15] Uriel Feige, Vahab S Mirrokni, and Jan Vondrak. Maximizing non-monotone submodular functions. *SIAM Journal on Computing*, 40(4):1133–1153, 2011.

[16] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[17] Naveen Garg, Goran Konjevod, and R Ravi. A polylogarithmic approximation algorithm for the group Steiner tree problem. In *SODA*, pages 253–259, 1998.

[18] Michel X Goemans and David P Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming. *Journal of the ACM (JACM)*, 42(6):1115–1145, 1995.

[19] Sudipto Guha and Samir Khuller. Approximation algorithms for connected dominating sets. *Algorithmica*, 20(4):374–387, 1998.

[20] F Hadlock. Finding a maximum cut of a planar graph in polynomial time. *SIAM Journal on Computing*, 4(3):221–225, 1975.

[21] David J. Haglin and Shankar M Venkatesan. Approximation and intractability results for the maximum cut problem and its variants. *IEEE Transactions on Computers*, 40(1):110–113, 1991.

[22] Subhash Khot, Guy Kindler, Elchanan Mossel, and Ryan O'Donnell. Optimal inapproximability results for MAX-CUT and other 2-variable CSPs? *SIAM Journal on Computing*, 37(1):319–357, 2007.

[23] Samir Khuller, Manish Purohit, and Kanthi K Sarpatwar. Analyzing the optimal neighborhood: Algorithms for budgeted and partial connected dominating set problems. In *SODA*, pages 1702–1713. SIAM, 2014.

[24] Ton Kloks. *Treewidth: computations and approximations*, volume 842. Springer, 1994.

[25] Tung-Wei Kuo, Kate Ching-Ju Lin, and Ming-Jer Tsai. Maximizing submodular set function with connectivity constraint: theory and application to networks. In *INFOCOM*, pages 1977–1985. IEEE, 2013.

[26] Rajeev Motwani and Prabhakar Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[27] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions-I. *Mathematical Programming*, 14(1):265–294, 1978.

[28] Slav Petrov. Image segmentation with maximum cuts. 2005.

[29] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *STOC*, pages 255–264. ACM, 2008.

[30] Neil Robertson and Paul D Seymour. Graph minors. III. Planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984.

[31] Roberto Solis-Oba. 2-approximation algorithm for finding a spanning tree with maximum number of leaves. In *ESA*, pages 441–452. Springer-Verlag, 1998.

[32] Chaitanya Swamy and Amit Kumar. Primal–dual algorithms for connected facility location problems. *Algorithmica*, 40(4):245–269, 2004.

[33] Sara Vicente, Vladimir Kolmogorov, and Carsten Rother. Graph cut based image segmentation with connectivity priors. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8. IEEE, 2008.

A   Dynamic program for constant tree-width graphs

The notion of tree decomposition and tree-width was first introduced by Robertson and Seymour [30]. Given a graph $G = (V, E)$, its tree decomposition is a tree representation $T = (\mathcal{B}, \mathcal{E})$, where each $b \in \mathcal{B}$ (called as a *bag*) is associated with a subset $B_b \subseteq V$ such that the following properties hold:

1. $\bigcup_{b \in B} B_b = V$.
2. For every edge $u, v \in E$, there is a bag $b \in \mathcal{B}$, such that $u, v \in B_b$.
3. For every $u \in V$, the subgraph T_u of T, induced by bags that contain u, is connected.

The width of a decomposition is defined as the size of the largest bag $b \in B$ minus one. Treewidth of a graph is the minimum width over all the possible tree decompositions. In this section, we show that the CMC problem can be solved optimally in polynomial time on graphs with constant treewidth t.

Notation. We denote the tree decomposition of a graph $G = (V, E)$ by $\mathcal{T} = (\mathcal{B}, \mathcal{E})$. For a given bag of the decomposition $b \in \mathcal{B}$, let B_b denote the set of vertices of G contained in b and V_b denote the set of vertices in the subtree of $\mathcal{T}$ rooted at b. As shown by Kloks [24], we may assume that $\mathcal{T}$ is nice tree decomposition, that has the following additional properties.

1. Any node of the tree has at most 2 children.
2. A node b with no children is called a leaf node and has $|B_b| = 1$.

3. A node b with two children c_1 and c_2 is called a join node. For such a node, we have $B_b = B_{c_1} = B_{c_2}$.
4. A node b with exactly one child c is either a forget node or an introduce node. If b is a forget node then $B_b = B_c \setminus \{v\}$ for some $v \in B_c$. One the other hand, if b is an introduce node then $B_b = B_c \cup \{v\}$ for $v \notin B_c$.

We now describe a dynamic program to obtain the optimal solution for the CMC problem. Let OPT denote the optimal solution. We first prove the following simple claim that helps us define the dynamic program variable.

Claim. For any bag $b \in \mathcal{B}$, the number of components induced by $OPT \cap V_b$ in G is at most t.

Proof. Consider the induced subgraph $G[OPT \cap V_b]$ and let C be one of its components. We observe that C has at least one vertex in B_b, i.e., $C \cap B_b \neq \phi$. Assume this is not true and $C \cap B_b = \phi$. Now consider an edge $e = (u, v)$ such that $u \in C$ and $v \in OPT \setminus C$. Such an edge is guaranteed to exist owing to the connectivity of $G[OPT]$. By our assumptions, $v \notin V_b$. This implies there is some bag b' not in the subtree of $\mathcal{T}$ rooted at b, that contains both u and v. But this in turn implies $u \in B_b$, a contradiction to the assumption $C \cap B_b = \phi$. Now, since each vertex in B_b belongs to at most one component, there can be at most $|B_b| \leq t$ components in $G[OPT \cap V_b]$. Hence, the claim.

For a given $b \in \mathcal{B}$, let $S_b = OPT \cap B_b$ be the set of vertices chosen by the optimal solution from the bag b. Further, let $P_b = (C_1, C_2 \ldots C_t)$ be a partition, of size t, of the vertices in S_b, such that each non-empty C_i (some of the C_i's could possibly be empty) is a subset of a unique component of the subgraph induced by $V_b \cap OPT$. We now define the variable of the dynamic program $M_b(P_b, S_b)$ in the following way: Consider the subgraph induced by V_b in G and let S be a subset of V_b with maximum cut $\delta_{G[V_b]}(S)$, such that every $C_i \in P_b$ is completely contained in a distinct component of $G[S]$. We set $M_b(P_b, S_b) = |\delta_{G[V_b]}(S)|$.

From this definition, it follows that the optimal solution can be obtained by computing $M_r(P_r = (S, \phi, \phi, \ldots, \phi), S)$, for every subset S of B_r, where r is the root bag of the tree $\mathcal{T}$ and picking the best possible solution. We now describe the dynamic program to compute the above variable $M_b(P_b, S_b)$ for a given bag b.

Case 1: Node $b \in \mathcal{T}$ is a leaf node. In this case, $B_b = V_b = \{v\}$, for some vertex v.

$$M_b((\{v\}, \phi, \ldots \phi), \{v\}) = 0$$
$$M_b((\phi, \phi, \ldots \phi), \phi) = 0$$

Case 2: Node $b \in \mathcal{T}$ is an introduce node. In this case, b has exactly one child node c and $B_b = B_c \cup \{v\}$, for some vertex v. Let $P_b = (C_1, C_2 \ldots C_t)$ be some partition of $S_b \subseteq B_b$. We compute $M_b(P_b, S_b)$ as follows.

If $v \notin S_b$, then

$$M_b(P_b, S_b) = M_c(P_b, S_b) + |\delta_{G[B_b]}(v, S_b)|$$

If $v \in C_i$ and v is adjacent to some vertex in $C_i \setminus \{v_i\}$ but not to any vertex in $C_j, j \neq i$, then

$$M_b(P_b, S_b) = M_c((C_1, C_2, \ldots, C_i \setminus \{v\}, \ldots, C_t), S_b \setminus \{v\}) + |\delta_{G[B_b]}(v, B_b \setminus S_b)|$$

In all other cases, we set

$$M_b(P_b, S_b) = -\infty$$

We now argue about the correctness of this case. First assume that $v \notin S_b$, that is v is not chosen into our solution. If $S \subseteq V_c$ is the set of vertices chosen into our solution so far, the total cut size increases by $|\delta_{G[V_c]}(v, S)|$, which we claim is equal to $|\delta_{G[B_b]}(v, S_b)|$. In other words, none of the edges incident on v are adjacent to any vertices in $S \setminus S_b$. Suppose for the sake of contradiction that there exists a vertex $w \in S \setminus S_b$ be such that $\{v, w\} \in E$. This implies that there exists a bag b', not in the subtree of b, that contains both v and w. This in turn implies that $w \in S_b$, which is a contradiction to the fact that $w \in S \setminus S_b$. Hence, in this case the total cut increases by $|\delta_{G[B_b]}(v, S_b)|$.

Now assume that $v \in S_b$, more specifically, let $v \in C_i$, for some i. From the above argument there are no edges between v and vertices in $S \setminus S_b$. Since we must have all vertices in C_i in a single component of $G[V_b]$ and all edges incident on v in $G[V_b]$ have the other end in B_b, v must have an edge to some vertex in $C_i \setminus \{v\}$. Further, since any C_i and C_j must belong to distinct components, they must not share any vertices. Thus, if v either has no edges to $C_i \setminus \{v\}$ or has an edge to some C_j, there is no feasible solution and we assign $-\infty$ to this variable. On the other hand if both these conditions are satisfied, our solution is valid and the increase in the cut size is $|\delta_{G[V_b]}(v, B_b \setminus S_b)|$.

Case 3: Node $b \in \mathcal{T}$ is a forget node. In this case, again, b has exactly one child node c with $B_b = B_c \setminus \{v\}$. Let $P_b = (C_1, C_2 \ldots C_t)$. It is easy to see that:

$$M_b(P_b, S_b) = \max \begin{cases} M_c(P_b, S_b) \\ M_c((C_1, C_2, \ldots, C_i \cup \{v\}, \ldots, C_t), S_b \cup \{v\}), \forall i \in [t] \end{cases}$$

Case 4: Node $b \in \mathcal{T}$ is a join node. In this case, b has two children c_1, c_2 with $B_b = B_{c_1} = B_{c_2}$. Let $P_b = (C_1, C_2 \ldots, C_t)$ be the partition of vertices in S_b such that each C_i belongs to a distinct component in $G[OPT \cap V_b]$. Similarly, define $P_{c_1} = (C_1^1, C_2^1 \ldots C_t^1)$ and $P_{c_2} = (C_1^2, C_2^2 \ldots C_t^2)$ as partitions of $S_{c_1} = S_b$ and $S_{c_2} = S_b$ respectively. Consider the construction of following auxiliary graph, that we refer to as "merge graph", denoted by M. For every C_i^1 and C_i^2, we have a corresponding vertex $v(C_i^1)$ and $v(C_i^2)$ respectively in M. Further if two sets C_i^1 and C_j^2 intersect, i.e., $C_i^1 \cap C_j^2 \neq \phi$, then we add an edge between $v(C_i^1), v(C_j^2)$ in M. It is easy to observe that for a given component C of M, the union of all subsets of vertices corresponding to the vertices of C must belong to the same component of $G[OPT \cap V_b]$. Thus in turn implies that there is a one to one correspondence between $C_i \in P_b$ and the components of M.

For a given partition P_b of S_b, we call two partitions P_{c_1} and P_{c_2} of S_{c_1} and S_{c_2} as "valid" if there is a one to one correspondence between $C_i \in P_b$ and the components of M, as described above. We now prove the following simple claim.

Claim. For any $S \subseteq V_b$, $\delta_{G[V_b]}(S, V_b \setminus S) = \delta_{G[V_{c_1}]}(S_1, V_{c_1} \setminus S_1) + \delta_{G[V_{c_2}]}(S_2, V_{c_2} \setminus S_2) - \delta_{G[V_b]}(S_b, B_b \setminus S_b)$, where $S_1 = S \cap V_{c_1}$ and $S_2 = S \cap V_{c_2}$

Proof. From the properties of tree decomposition, it follows that for any two vertices $u \in V_{c_1} \setminus B_b$ and $w \in V_{c_2} \setminus B_b$, $uw \notin E$. Further all the edges in $\delta(V_{c_1} \setminus B_b)$ and $\delta(V_{c_1} \setminus B_b)$ are

incident on the vertices in B_b. Thus we have the following equation,

$$\begin{aligned}
\delta_{G[V_b]}(S, V_b \setminus S) =&\delta_{G[V_{c_1}]}(S_1 \setminus B_b, V_{c_1} \setminus S_1) + \delta_{G[V_{c_2}]}(S_2 \setminus B_b, V_{c_2} \setminus S_2) + \delta_{G[V_b]}(S_b, B_b \setminus S_b) \\
=&(\delta_{G[V_{c_1}]}(S_1 \setminus B_b, V_{c_1} \setminus S_1) + \delta_{G[V_b]}(S_b, B_b \setminus S_b)) \\
&+ (\delta_{G[V_{c_1}]}(S_2 \setminus B_b, V_{c_2} \setminus S_2) + \delta_{G[V_b]}(S_b, B_b \setminus S_b)) - \delta_{G[V_b]}(S_b, B_b \setminus S_b) \\
=&\delta_{G[V_{c_1}]}(S_1, V_{c_1} \setminus S_1) + \delta_{G[V_{c_2}]}(S_2, V_{c_2} \setminus S_2) - \delta_{G[V_b]}(S_b, B_b \setminus S_b)
\end{aligned}$$

We can now compute the dynamic program variable, in this case, as follows.

$$M_b(P_b, S_b) = \max_{\substack{(P_{c_1}, P_{c_2}) \text{ are valid} \\ \text{with respect to } P_b}} M_{c_1}(P_{c_1}, S_b) + M_{c_2}(P_{c_2}, S_b) - |\delta_{G[V_b]}(S_b, B_b \setminus S_b)|$$

B Maximum Leaf Spanning Tree

In the Maximum Leaf Spanning Tree problem, we are given an undirected graph $G = (V, E)$ and the goal is to find a spanning tree that has a maximum number of leaves. We now show that this well studied problem is a special case of the Connected Submodular Maximization problem. Consider maximizing the submodular function $f(S) = |\{v \mid v \in \mathbb{N}(S) \setminus S\}|$ where $\mathbb{N}(S)$ is the set of vertices that have a neighbor in S. Now, it is easy to observe that there is a tree with L leaves if and only if there is a connected set S such that $f(S) = L$. Further, we show that without loss of generality for any solution S to Connected Submodular Maximization problem, we have that $S \cup \mathbb{N}(S) = V$ and hence the corresponding tree is spanning. Suppose S is a feasible solution and $V \neq S \cup \mathbb{N}(S)$, then there must exist an edge (u, v) such that $u \in \mathbb{N}(S) \setminus S$ and $v \notin S \cup \mathbb{N}(S)$. Now $S' = S \cup \{u\}$ is also a feasible solution and $f(S) = f(S')$.

C High Connectivity

We observe that the Connected Submodular Maximization has a constant factor approximation algorithm if the graph has high connectivity. Let $S \subseteq V$ be a random set of vertices such that every vertex v is chosen in S independently with probability $\frac{1}{2}$. As shown by Feige et al. [15], $\mathbb{E}[f(S)] \geq (\frac{1}{4})f(OPT)$ where the $f(OPT)$ is the maximum value of f. Further, as shown by Censor-Hillel et al. [5] if the graph G has $\Omega(\log n)$ vertex connectivity, the set S obtained above is connected with high probability. Hence, S is a $\frac{1}{4}$ approximate solution to the Connected Submodular Maximization problem.