Clique Cover on Sparse Networks

Mathieu Blanchette* Ethan Kim* Adrian Vetta†

Abstract

We consider the problem of edge clique cover on sparse networks and study an application to the identification of overlapping protein complexes for a network of binary protein-protein interactions. We first give an algorithm whose running time is linear in the size of the graph, provided the treewidth is bounded. We then provide an algorithm for planar graphs with bounded branchwidth upon which we build a PTAS for planar graphs. Empirical studies show that our algorithms are both efficient and practical on actual simulated and biological networks, and that the clique covers obtained on real networks yield biological insights.

1 Introduction

Given a graph $G = (V, E)$, the *(edge) clique cover* problem asks for a smallest collection Q of cliques in G such that every edge[1] of G is covered by at least one clique in Q. If each edge is required to be covered by precisely one clique in Q, the problem is known as the *clique partition* problem. These problems have a large number of applications in diverse disciplines. Here we consider one prominent example from computational biology. Protein-protein interaction (PPI) networks are traditionally modelled as graphs whose vertices represent proteins, and vertices are joined by an edge if and only if the two corresponding proteins interact with each other. Recent studies [3], however, have discovered a number of interactions involving more than 2 proteins. A protein complex is a set of interacting proteins, involving anywhere from 2 to 50 proteins, and forming cliques (or, at least, very dense subgraphs) in the network. A number of algorithms have been proposed to detect these complexes by various clique or dense-subgraph discovery algorithms [19, 28]. Because a protein can often belong to different complexes, cliques typically overlap, and thus computational biologists have extended their algorithms to allow for overlapping cliques [1, 3, 22]. To our knowledge, however, none of these algorithms are guaranteed to exactly solve the problem they are designed for.

As well as graph theoretic aspects, the clique cover problem has been studied extensively from the standpoint of computational complexity. In particular, there are numerous studies concerning approximability and fixed-parameter tractability. In terms of approximation, clique cover is NP-hard in general [25], and even when the input graph is planar [8] or has bounded degree [17]. Furthermore, Lund and Yannakakis have shown that clique cover is not approximable within a factor of $|V|^{\epsilon}$ for some $\epsilon > 0$ unless $P = NP$ [23], thereby removing the hope of good approximation algorithms in the general case. In the case of clique *partition* problem, the problem has also been shown to be NP-complete for various restricted classes of graphs [7, 13, 14, 18].

A parameterized problem is *fixed-parameter tractable* (FPT) if it can be solved in $f(k) \cdot |I|^{O(1)}$ time, where f is a computable function depending on some parameter k, independent of the input size $|I|$. Recently, Gramm *et al.* [15] showed that the clique cover problem is FPT when the size of the cover is chosen for the parameter k. Similarly, Mujuni and Rosamond [24] have shown that the clique partition problem is FPT with the output size chosen as the parameter. These algorithms run in polynomial time in the input size but exponential time in the number of cliques in the solution. As a result, these algorithms are well suited for dense graphs where a few cliques can cover the entire graph, but they are not suitable for sparse graphs that require a large number of small cliques in the solution.

In this paper, we fill the gap by designing efficient algorithms for such sparse graphs. We first introduce some definitions and related results in Section 2. Then, in Section 3, we will design an exact algorithm for clique cover when the input graph has bounded treewidth. Section 4 discusses the problem restricted to planar graphs, and we provide a polynomial time approximation scheme. Finally, we show the performance of our algorithm from experimental studies in Section 5. In particular, our algorithm shows efficient and practical running time when computing for both real and simulated biological networks. Furthermore, our PTAS for

*McGill Centre for Bioinformatics, McGill University, Canada. Email: blanchem@mcb.mcgill.ca, ethan@cs.mcgill.ca.

†Department of Mathematics & Statistics, McGill University, Canada. Email: vetta@math.mcgill.ca

[1]Note that one can require each vertex (instead of each edge) to be covered by the clique cover. This vertex clique cover problem is equivalent to the well-studied graph colouring problem.

planar graphs shows a clear trade-off between approximation ratio and the running time when tested against random planar graphs.

2 Preliminaries

In this paper, we study the clique cover problem for sparse networks. Where possible, we shall discuss how the algorithms for the clique cover problem can be modified to solve the clique partition problem. Various measures of network sparsity have been proposed in the past, with possibly the best known being treewidth.

DEFINITION 2.1. [26] *Tree decomposition of a graph $G = (V, E)$ is a pair $(X = \{X_i | i \in I\}, T = (I, F))$ where each node $i \in I$ is associated with a set of vertices $X_i \subseteq V$, such that*

(1) Each vertex belongs to at least one node: $\bigcup_{i \in I} X_i = V$.

(2) Each edge is induced by at least one node: $\forall (v, w) \in E$, there is an $i \in I$ with $v, w \in X_i$.

(3) For each $v \in V$, the set of nodes $\{i \in I | v \in X_i\}$ induces a subtree of T.

The width of a tree decomposition (X, T) is defined as $max_{i \in I} |X_i| - 1$, and the *treewidth* of a graph G, denoted $tw(G)$, is the minimum width over all tree decompositions of G. In general, it is NP-Complete to determine the treewidth of a graph [2]. However, when k is fixed, graphs with treewidth k can be recognized, and width k tree decompositions can be constructed, in linear time [6].

From the empirical studies of our input PPI data (shown in Section 5, Table 1), the treewidth of PPI networks is often small compared to the network sizes. We thus assume, where applicable, that the input graph has a bounded treewidth, and its optimum tree decomposition can be constructed efficiently. Furthermore, for simplicity of discussions, we assume that the decomposition tree T admits a *nice* structure as defined below.

DEFINITION 2.2. [20] *A tree decomposition (X, T) is called nice if the tree T is rooted, and for each node $i \in I$, one of the following holds:*

1. *Leaf: node i is a leaf of T, and $|X_i| = 1$.*
2. *Join: node i has exactly two children j_1 and j_2 such that $X_i = X_{j_1} = X_{j_2}$.*
3. *Introduce: node i has exactly one child j, and $X_i = X_j \cup \{v\}$.*
4. *Forget: node i has exactly one child j, and $X_j = X_i \cup \{v\}$.*

It is not hard to see that if $tw(G) \leq k$, then G also admits a nice tree decomposition of width $\leq k$, with $O(n)$ tree nodes: given an arbitrary decomposition tree T, one can repeatedly split each node X_i until all nodes satisfy the conditions above.

Another closely related graph parameter is *branchwidth*.

DEFINITION 2.3. [27] *A branch decomposition (T, ϕ) of a graph G is characterized by a ternary tree[2] T, and a bijection ϕ from the leaves of T onto the edges of G.*

Let e be a tree edge in T. Removing e from T partitions into T_1 and T_2, and this partition induces a partition of edges in G, called an *e-separation*, associated with the leaves of T_1 and T_2. The set of vertices in G that are shared by both G_1 and G_2 is called the *middle-set* of e, and the *width* of this separation is the number of vertices in the middle-set.

Given a branch decomposition (T, ϕ), the width of this branch decomposition is the maximum width over all e-separations in T, and the branchwidth of G, denoted $bw(G)$, is the minimum width over all branch decompositions. It is well known that the branchwidth is closely related to the treewidth of graph [27]: $bw(G) \leq tw(G) + 1 \leq \frac{3}{2} bw(G)$. For planar graphs, Fomin and Thilikos gave an upperbound on the branchwidth:

THEOREM 2.1. [12] *For any planar graph G, $bw(G) \leq \sqrt{4.5n} \approx 2.122\sqrt{n}$.*

3 Clique Cover for Graphs with Bounded Treewidth

In this section, we design a dynamic programming algorithm for finding a minimum clique cover for a graph G where a *nice* tree decomposition (X, T) is given. Let k denote the width of that decomposition. First, let us define $E(X_i)$ to be the set of edges in the subgraph induced by the vertices in X_i. Furthermore, we let V_i denote the *union* of all vertices in X_i *and* its descendent nodes. Similarly, let G_i denote the subgraph of G induced by the vertices V_i. Finally, for some $v \in V(G)$, let $\delta(v)$ denote the set of edges that are incident to v in G.

We shall in fact design an algorithm for a generalization of the clique cover problem, where we are given a subset S of edges that are *already covered*, i.e., our solution need not cover S, but may use these edges in the cliques. Then, the original clique cover problem is a

[2] A tree T is a ternary tree if every non-leaf node has degree 3.

special case where $S = \emptyset$. Since our dynamic programming is formulated around the decomposition tree T, we often speak of a subgraph G_i where a certain subset S of edges is already covered, denoted by $G_i(S)$. Now we can define a cost function:

$$C_i(S) = \text{minimum size of clique cover for } G_i \text{ where } S \text{ is already covered.}$$

Then, our final solution is precisely $C_r(\emptyset)$ where r is the root of T. Our dynamic programming algorithm will proceed from the leaves of T up to its root, computing, for each node X_i, the value of $C_i(S)$ for every possible subset S of $E(X_i)$. Depending on the type of node, $C_i(S)$ is computed differently.

LEAF NODE. Suppose i is a leaf node. Then, by the definition of nice tree decomposition, $|X_i| = 1$, and thus $C_i(S) = 0$ for all $S \subseteq E(X_i)$, trivially.

FORGET NODE. Suppose i is a forget node. Then, it has one descendant $X_j = X_i \cup \{v\}$ for some unique vertex v.

LEMMA 3.1. *If i is a Forget node, then for any subset $S \subseteq E(X_i)$, $C_i(S) = C_j(S)$.*

Proof. Note that since $X_i \subset X_j$, the corresponding graphs G_i and G_j are the same. Furthermore, since v is not in X_i, $S \cap \delta(v) = \emptyset$. Therefore, $S \cap E(X_i) = S \cap E(X_j)$ and we have $C_i(S) = C_j(S)$. □

INTRODUCE NODE. Suppose i is an introduce node. Then, it has one descendant node X_j such that $X_i = X_j \cup \{v\}$ for some unique vertex v. Consider an arbitrary clique cover $\mathcal{W}$ for $G_i(S)$. Since the cliques in $\mathcal{W}$ can be partitioned into Q_v = cliques containing v and $Q_{\overline{v}} = \mathcal{W} - Q_v$, the following recurrence relation holds for each introduce node.

LEMMA 3.2. *If i is an Introduce node, then for any subset $S \subseteq E(X_i)$:*

$$C_i(S) = min\{\ |Q_v| + C_j(S \cup E(Q_v))\ :\ Q_v \text{ is a clique cover for } \delta(v) - S\ \}$$

Proof. Let $\mathcal{W}$ be a clique cover for $G_i(S)$. We partition the cover $\mathcal{W} = Q_v \cup Q_{\overline{v}}$ as defined, and consider the edgeset in Q_v. Since these edges are covered by Q_v, $Q_{\overline{v}}$ needs only cover $G_i - (S \cup Q_v)$. Moreover, since the cliques in $Q_{\overline{v}}$ do not contain v, $Q_{\overline{v}}$ is a cover for $G_j - (S \cup Q_v)$. Therefore, $|Q_{\overline{v}}| \geq C_j(S \cup Q_v)$, and the lemma follows. □

To compute $C_i(S)$, we need to consider all possible clique covers, Q_v, for $\delta(v) - S$. Since $|X_i| \leq k$, this is the number of ways to partition k vertices, and is given by the kth Bell number $B(n) \leq k!2^k$.

JOIN NODE. Finally, suppose i is a join node. Then, it has two children nodes j_1 and j_2 such that $X_i = X_{j_1} = X_{j_2}$, and $G_i = G_{j_1} \cup G_{j_2}$. Therefore, a clique cover for G_i contains cliques that belong to G_{j_1} or G_{j_2}. We need to ensure not to double count cliques that belong in both G_{j_1} and G_{j_2}.

LEMMA 3.3. *Let $S \subseteq E(X_i)$ be the set of already covered edges, and let $R = E(X_i) - S$ be the edges to be covered. Then,*

$$C_i(S) = min\{\ C_{j_1}(S \cup R_2) + C_{j_2}(S \cup R_1)\ :\ \forall R_1 \subseteq R \text{ and } R_2 = R - R_1\}$$

Proof. Assuming S is already covered, let $\mathcal{W}$ be a minimum clique cover for $G_i(S)$ with cost $C_i(S)$. By definition, any clique that belongs to both G_{j_1} and G_{j_2} must also belong to X_i. Thus, the cliques in $\mathcal{W}$ can be partitioned as $\mathcal{W} = Q_1 \cup Q_2 \cup Q_3$, where

$$Q_1 = \{q \in \mathcal{W} \mid q \subseteq V_{j_1} \text{ and } q \not\subseteq X_i\}$$
$$Q_2 = \{q \in \mathcal{W} \mid q \subseteq V_{j_2} \text{ and } q \not\subseteq X_i\}$$
$$Q_3 = \{q \in \mathcal{W} \mid q \subseteq X_i\}.$$

Thus, $|\mathcal{W}| = |Q_1| + |Q_2| + |Q_3|$. Now, the edgeset R can be partitioned to $R = R_1 \cup R_2$ such that:

$$R_1 = \{e \in R \mid e \text{ covered by } Q_1 \text{ or } Q_3\ \}$$
$$R_2 = \{e \in R \mid e \text{ covered only by } Q_2\ \} = R - R_1$$

By definition, $Q_1 \cup Q_3$ needs to cover the edges in R_1. Furthermore, Q_3 needs to cover the edges in $G_{j_1} - E(X_{j_1})$, and thus $|Q_1 \cup Q_3| \geq C_{j_1}(S \cup R_2)$. On the other hand, the cliques in Q_2 only need to cover the edges in R_2 together with $G_{j_2} - E(X_{j_2})$, and thus $|Q_2| \geq C_{j_2}(S \cup R_1)$, and the result follows. □

Therefore, we can compute $C_i(S)$ for any given subset $S \subseteq E(X_i)$. Observe that the recurrence relation looks at all possible bipartitions of R. Since the number of edges in $E(X_i)$ is at most $\binom{k}{2}$, we need to check at most $2^{\binom{k}{2}}$ different partitions of R. Once the bipartition of R is fixed, it takes constant time to look up the values from C_{j_1} and C_{j_2}.

Note that, while these recurrence relations calculate the size of clique covers, they are also constructive: a little bookkeeping at each node will allow us to construct the optimal cover at the root node.

3.1 Running time. For each node $i \in I$, we compute $C_i(S)$ for every $S \subseteq E(X_i)$. Since $tw(G) = k$, each node

contains at most k vertices. Let ρ denote the maximum number of edges induced in any node. Then we need to consider 2^ρ cases. Then, for each fixed $S \subseteq E(X_i)$, we carry out one of the four recurrence relations. Leaf nodes and Forget nodes can be computed in constant time. An Introduce node can be computed in $O(B(k)\cdot k)$ time, where $B(k)$ is kth Bell number. Finally, a Join node takes $O(2^\rho)$ to compute. Therefore, the dynamic programming algorithm takes $2^\rho \cdot max\{B(k)\cdot k, 2^\rho\} \cdot O(n) = O(4^\rho n)$ time overall. While ρ can be as large as $\binom{k}{2}$ in theory, this is rarely the case as shown in our experimental tests (see Section 5, Table 1).

THEOREM 3.1. *There is a linear time algorithm for computing minimum clique cover for graphs with fixed treewidth k.*

3.2 Modifications for clique partition problem. It is straightforward to modify the above algorithm to solve the clique partition problem: instead of assuming that edges already covered can be re-used to form other cliques, we simply delete those edges and solve for remaining edgeset. If we redefine the cost function $C_i(S)$ to be the size of the minimum clique partition for the graph $G_i - S$, the same recurrence holds for each node type. The only difference is when, at an Introduce node, finding a local solution for $\delta(v)$, we look for a clique partition rather than a cover.

4 Planar Clique Cover

In this section, we study the clique cover problem restricted to planar graphs, and present a PTAS for planar graphs. While planar graphs are possibly the most restricted class of interest for the clique cover problem (the largest clique being just K_4), the problem remains NP-hard [30]. Furthermore, as treewidth is unbounded in planar graphs [16, 26], simply applying our algorithm from Section 3 would result in exponential running time, and since tree decompositions do not depend on planarity of the input graph, and it may be difficult to specialize the algorithm to planar graphs.

Instead, we shall design an exact polynomial time algorithm for planar graphs with bounded *branchwidth*. As we shall see, the algorithm runs in $O(2^k n)$ time with k being the branchwidth, and since $bw(G) \leq \sqrt{4.5n}$ when G is planar, this would be the first subexponential algorithm for the clique cover problem on planar graphs.

Then, we will use our exact algorithm to construct a polynomial time approximation scheme. Baker [4] has proposed a divide-and-conquer technique to design approximation schemes for various optimization problems on planar graphs. We will show that her technique can be applied to the planar clique cover problem, using our exact algorithm as a subroutine, resulting in a $(1+\epsilon)$ approximation algorithm.

4.1 Clique cover for planar graphs with bounded branchwidth. As with its counterpart, treewidth, it is NP-complete to determine if a graph has a branch decomposition of width at most k in general, but this decomposition can be found in linear time when k is fixed. We thus assume that the input graph G is given together with a branch decomposition (T, ϕ) of width at most k. Now pick an arbitrary edge e of T, and subdivide it to create a root node r. Then each tree node X is associated with a subset of edges in $E(G)$, namely the leaf nodes of the subtree rooted at X. We let $E(S)$ denote the edges in the subgraph induced by a subset of vertices S.

Define the middle-set of X, denoted by $mid(X)$, to be the middle-set of the edge between X and its parent node. Since the root node r has no parent, set $mid(r) = \emptyset$. Then, we create a table $W_X[\cdot]$ indexed by a subset $\mathcal{F}$ of edges as follows:

$$W_X[\mathcal{F}] = \text{minimum clique cover for edges in } X \text{ with } \mathcal{F} \text{ already covered.}$$

Since $mid(X)$ is a cutset in G, we can paste together solutions from each subproblem by computing only the entries $W_X[\mathcal{F}]$ where $\mathcal{F}$ is a subset of edges in $E(mid(x))$.

Before describing the recurrence relation for this table, we study the middle-set of three adjacent edges. Consider the sphere-cut branch decomposition for planar graphs, as studied by Dorn *et al.* [11]; here, each middle-set defines a closed curve (*noose*) on the planar embedding of the input graph that intersects only the vertices in the middle-set. Let X be a tree node with two children nodes X_1 and X_2. The three edges adjacent to X define 3 middle-sets which we denote by O_P, O_L, O_R for parent edge, left and right child edge, respectively. Since $O_R - (O_P \cup O_L) = \emptyset$ and $O_L - (O_P \cup O_R) = \emptyset$, the vertices of $O_P \cup O_L \cup O_R$ can be partitioned as follows:

- Portal vertices $P = O_L \cap O_R \cap O_P$
- Intersection vertices $I = O_L \cap O_R - P$
- Symmetric Difference vertices $D = O_P - (P \cup I)$

LEMMA 4.1. *The table $W_X[\mathcal{F}]$ can be calculated as*

$$W_X[\mathcal{F}] = min\{W_{X_1}[\mathcal{F} \cup F_2] + W_{X_2}[\mathcal{F} \cup F_1] \ : \ F_1 \cup F_2 \text{ is a partition of } E(I \cup P)\}$$

Proof. For an arbitrary clique cover for X with $\mathcal{F}$ already covered, consider the cliques covering the edges

$E(I \cup P)$. Observe that, by planarity of G, any clique intersecting with $I \cup P$ only contains either vertices of X_1 or vertices of X_2. Therefore, we can partition the edges in $E(I \cup P)$ into F_1 and F_2, where F_1 is to be covered by cliques in X_1, and F_2 is covered by cliques in X_2. For each partition, the solution from the subproblem $W_{X_1}[\mathcal{F} \cup F_2]$ together with the solution from $W_{X_2}[\mathcal{F} \cup F_1]$ gives the solution for $W_X[\mathcal{F}]$. Since we consider all possible partitions of $E(I \cup P)$, the formula follows. □

The algorithm runs in a bottom-up manner. Observe that to compute each state $\mathcal{F}$ of a node, we need to consider all bipartitions of $E(I \cup P)$. Since G is planar, and $I \cup P \subseteq mid(X_1)$, $|E(I \cup P)| = O(k)$, and thus there are $2^{O(k)}$ such partitions. Moreover, to compute for all states $\mathcal{F}$, we consider $2^{O(k)}$ subsets of edges in $E(mid(X))$. Altogether, the above dynamic programming algorithm runs in $2^{O(k)}O(n)$ time.

LEMMA 4.2. *There is a linear time algorithm to compute a minimum clique cover for planar graphs with bounded branchwidth.*

4.2 Modifications for clique partition problem. As with our treewidth-based algorithm, the above algorithm can be easily modified to solve the clique partition problem: rather than assuming $\mathcal{F}$ is already covered, one can simply delete those edges and solve for the remaining edges.

4.3 Baker's technique on planar graphs. Baker [4] has proposed a general approach to design approximation algorithms for various NP-hard problems on planar graphs. Here, we show how this technique, together with our exact algorithm in Section 4.1, results in a $(1+\epsilon)$ approximation algorithm.

Baker's technique is a divide-and-conquer approach, where the input graph is decomposed into *layers* of subgraphs defined by the distance from a chosen vertex. Applying this technique to the planar clique cover problem, we construct the following scheme.

1. Arbitrarily choose a vertex r of G and execute a breadth-first search on G, starting at r.
2. Let $k = \lceil 2/\epsilon \rceil$. For $i = 0, 1, \ldots, k-1$ and $j = 0, 1, 2 \ldots$, let G_{ij} denote the subgraph of G induced by the vertices at levels $jk+i$ through $(j+1)k+i$, and compute a minimum clique cover C_{ij} for G_{ij}.
3. For $i = 0, 1, \ldots, k-1$, let $C_i = \bigcup_j C_{ij}$, and return whichever of $C_0, \ldots, C_{k-1}$ that has the least weight.

LEMMA 4.3. *A solution from the above approximation scheme has weight at most* $(1+\epsilon)$ *OPT.*

Proof. Let Q^* denote the optimum solution, and given some congruence class $i \mod k$, let us assume that the above approximation scheme divides the problem into m pieces. Since the solution for each piece is solved exactly, we have

$$|C_i| = \sum_{j=1}^{m} |C_{ij}| \ \leq \ \sum_{j=1}^{m} |Q^*_{ij}|,$$

where Q^*_{ij} denotes the optimum solution restricted to the graph G_{ij}. We therefore need to compare the two values $\sum_{j=1}^{m} |Q^*_{ij}|$ and $|Q^*|$. To compare these, consider the number of cliques in Q^* that contain vertices at levels $i \mod k$. For planar graphs, each clique belongs to at most 2 consecutive levels. Therefore, for at least one value of i, $0 \leq i \leq k$, there are at most $\lceil \frac{2}{k} \rceil |Q^*|$ cliques containing a vertex at level $i \mod k$. Since these cliques are double counted in the sum $\sum_{j=1}^{m} |Q^*_{ij}|$, we have

$$\sum_{i=1}^{m} |Q^*_i| \leq (1 + \frac{2}{k})|Q^*|$$

and it follows that $|C_i| \leq (1 + \frac{2}{k})|Q^*|$. □

According to Tamaki's theorem [29], a graph has a branchwidth at most the radius of the face-incidence graph[3] of G. Since the face-incidence graph of the subgraph G_{ij} has a bounded radius, each G_{ij} is a planar graph with bounded branchwidth. Therefore, our algorithm from Lemma 4.2 can compute the minimum clique cover for the subgraph in each layer.

Recall that our dynamic programming algorithm for graphs with branchwidth $\leq k$ runs in time $2^{O(k)}O(n)$. Due to Theorem 2.1, directly applying this algorithm to planar graphs gives an *exact* solution in time $2^{O(\sqrt{n})}O(n)$. On the other hand, the PTAS using Baker's technique involves decomposing the graph into n/k layers, and solving each piece exactly in time $2^{O(k)}O(n/k)$. Trying for every congruent classes between 1 and $k-1$, the overall algorithm takes $k \cdot \lceil \frac{n}{k} \rceil \cdot 2^{O(k)}O(n/k) \approx 2^{O(k)}O(\frac{n^2}{k})$ to obtain a solution of value at most $(1 + \frac{2}{k})OPT$. Therefore, while our PTAS provides an approximation with varying degree of approximation ratio, if one wants to get a solution any closer than $(1 + \frac{1}{\sqrt{n}})OPT$, one may be better off running our dynamic programming algorithm directly to the graph to obtain an exact solution. This trade-off is clearly shown empirically in Table 2.

THEOREM 4.1. *There is a polynomial time approximation scheme for planar clique cover.*

[3] Given a plane-embedded graph G, the face-incidence graph $\tilde{G} = (\tilde{V}, \tilde{E})$ consists of vertices $\tilde{V}$ from faces of G, and two vertices in $\tilde{V}$ are joined by an edge if and only if the corresponding faces in G share a vertex.

	n	m	tw	ρ	tree decomp. (hh:mm)	clique cover algo. (hh:mm)	# of cliques
Krogan	323	742	11	21	2:13	6:08	482
PAM	300	634	14	29	2:41	9:21	421
DM	300	794	21	43	4:13	17:47	583

Table 1: Performance of treewidth-based exact clique cover algorithm; we show the treewidth of each graph, maximum # of edges per tree node (ρ), time taken to compute an optimal tree decomposition and a clique cover, and size of the solution.

5 Experimental Studies

We have implemented the described algorithms and tested them against both real biological PPI data and simulated data[4]. As a comparison, we considered Gramm *et al.*'s algorithm [15] for the decision problem version of clique cover: given the size of clique cover k as an input parameter, their algorithm works by first applying a set of reduction rules to reduce the problem instance, and then using a search tree algorithm on the reduced instance, in time exponential in k. While their algorithm works well in cases where the solution size is small, it would perform poorly against our test data which contains several hundreds of cliques. This is mainly because their reduction rules did not significantly decrease the size of our input graphs (especially biological networks): the solutions for the reduced instances would still contain a large number of cliques, resulting in inefficient running time for the search tree algorithm. As their input parameter is the size of the minimum clique cover, it motivates our development of approaches using edge sparsity for these networks.

5.1 Simulated and biological networks. Each of our algorithms was tested on both simulated and actual biological networks. First, Krogan *et al.* [21] obtained an extensive dataset on protein interactions in yeast. Taking the largest connected component from the dataset, with 323 vertices and 742 edges, we formed a model network, G_{Krogan}. Various studies have shown that PPI networks exhibit the properties of scale-free networks [5]. Many generative models for scale-free networks have also been proposed, and we used the two most frequently used models [5, 9] to create test graphs for our algorithm: (1) preferential attachment model (denoted by G_{PAM}), and (2) duplication model (denoted by G_{DM}). In both cases, we set the parameters of generative models so that the resulting networks show similar characteristics to that of real PPI data; density of $|E| \approx 2|V|$, and degree distribution $P(k) \sim k^{-\gamma}$ where $\gamma \approx 1.7$.

To investigate the behavior of our algorithms on denser graphs, we generated a set of partial k-trees. Recall that a k-tree is a maximal graph with treewidth k such that no edge can be inserted without increasing its treewidth, and a graph is a partial k-tree if it is a subgraph of a k-tree. The partial k-trees of given treewidth have been generated by first generating a k-tree, and randomly removing edges to obtain desired edge density. Since the generation process of k-trees is similar to that of the preferential attachment model, this allows us to create graphs with higher density than G_{PAM} while preserving the treewidth bound.

5.2 Performance of the treewidth and branchwidth based algorithms. Table 1 reports results obtained on real and simulated biological networks. Both Krogan's PPI network and simulated networks exhibit relatively low treewidths for their size. Figure 1 gives a more comprehensive view of the running times from empirical testing. As expected, the running time increases linearly with n for graphs with fixed treewidths (Figure 1(a)), but exponentially with treewidth k but with fixed graph sizes (Figure 1(b)). Running times for partial k-trees (Figures 1(c) and (d)) follow the same trends, although they are somewhat higher due to the higher edge density.

While our branchwidth-based exact algorithm was designed for planar graphs, it is easy to modify the algorithm to handle non-planar graphs with bounded branchwidth (but at the expense of higher time complexity due to non-planarity). Figure 2 shows that, in practice, the running time of the treewidth-based algorithm grows slower than that of the branchwidth-based algorithm, allowing it to handle graphs with larger treewidths.

5.3 Performance of PTAS for planar graphs. To test our PTAS on planar graphs, we generated a

[4] http://www.cs.mcgill.ca/∼ethan/work/cliquecover

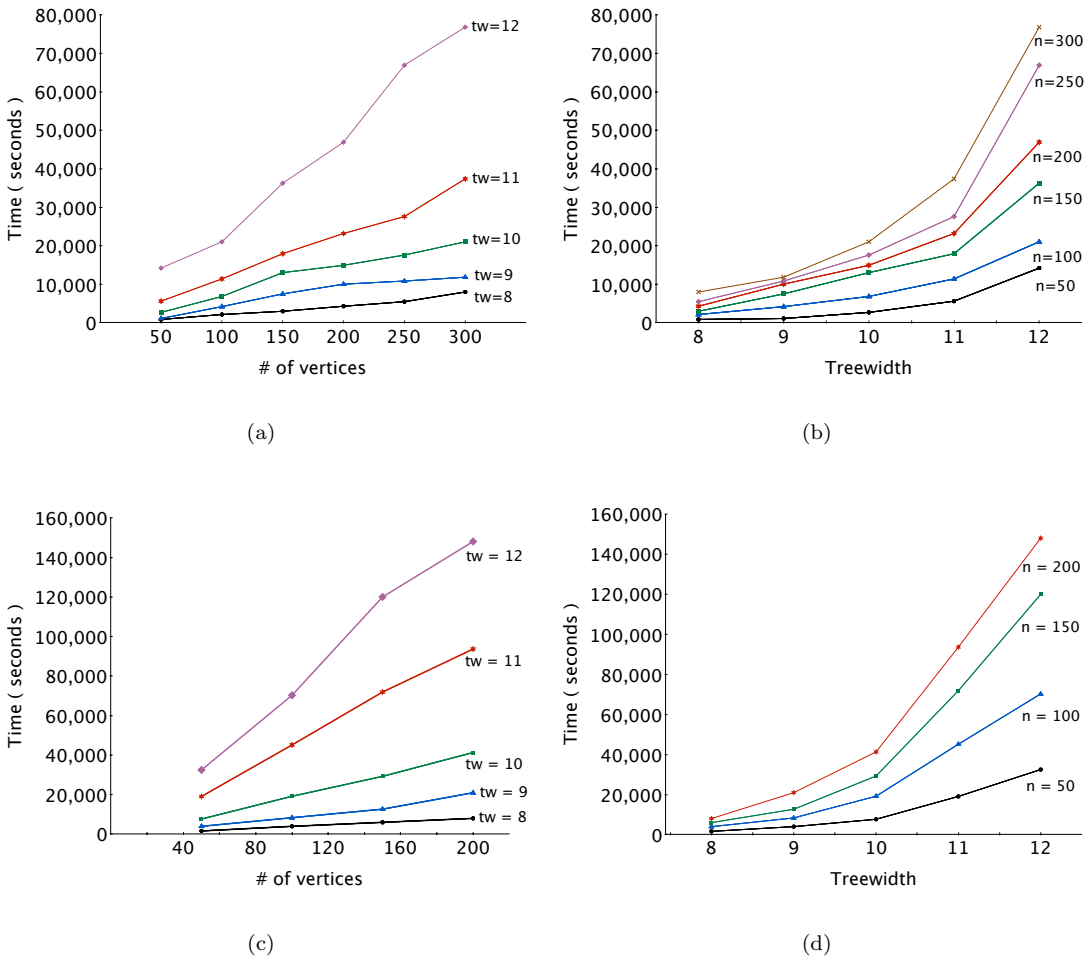
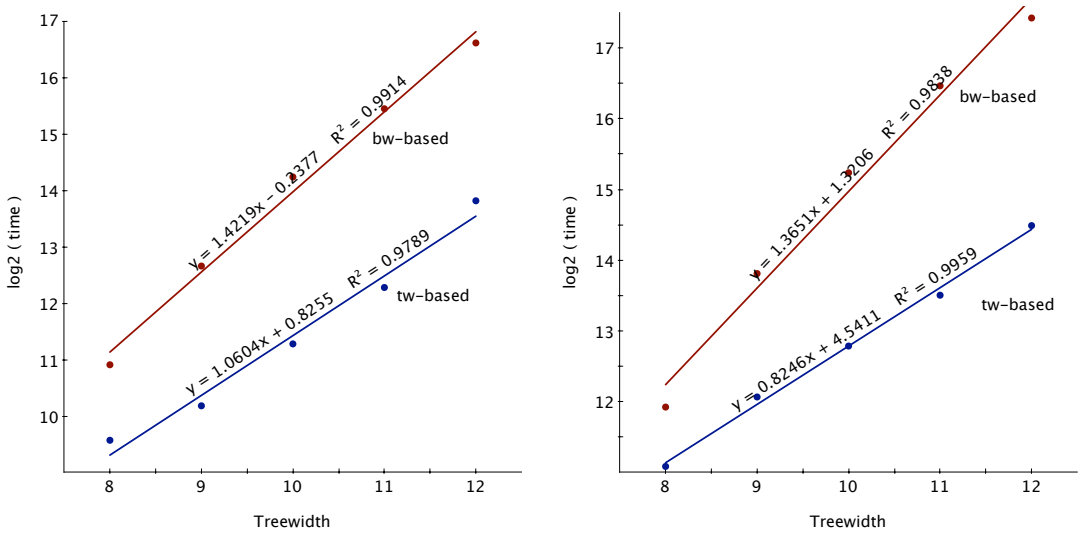


Figure 1: Performance of the treewidth-based algorithm on simulated networks: (a) scale-free networks (PAM) with fixed treewidth; (b) scale-free networks (PAM) with fixed graph size; (c) partial k-trees with fixed treewidth; (d) partial k-trees with fixed graph size.



(a) (b)

Figure 2: Performance comparison of treewidth-based algorithm vs. branchwidth-based algorithm on scale-free networks; y-axes are shown in log scale to exemplify the difference in exponents in the running time. (a) scale-free networks with 50 vertices; (b) scale-free networks with 100 vertices; Data points with the same treewidth values in each chart are taken from the same network.

set of random planar graphs using the simple algorithm by Denise and Vasconcellos [10]. Then, we ran both our exact branchwidth-based algorithm and the PTAS with varying values of ϵ to explore the trade-off between running time and quality of the solution. As shown in Table 2, the promised approximation ratios are almost exactly realized and substantial speed-ups are obtained for relatively large values of ϵ, compared to the exact branch-decomposition based algorithm. The running time increases exponentially with $1/\epsilon$, and the exact algorithm starts to become faster than the PTAS when ϵ becomes sufficiently small.

ϵ	# of cliques	time (hh:mm:ss)
0.5	159	00:12:08
0.2	124	00:39:17
0.1	116	02:36:18
0.05	113	03:53:42
OPT	109	02:46:31

Table 2: Performance of PTAS for planar graph with $n = 200, m = 407, tw = 10$. OPT was obtained using the branch-decomposition based algorithm.

5.4 Analysis of the clique cover of a biological network. When executed on the yeast protein-protein interaction network of Krogan *et al.* [21], our treewidth-based algorithm finds a clique cover that includes 93 cliques of size 5 or more. While the PPI data may not admit a unique clique cover on the network, we manually verified that most discovered cliques correspond to known complexes, such as the RNA polymerase II, the RSC complex, the mediator complex, and the 20S proteasome. In addition, three highly overlapping complexes, SWR1 (a chromatin remodelling complex), NuA4 (a histone acetyltransferase complex), and INO80 (another chromatin remodelling complex) are correctly identified, despite the fact that SWR1 and NuA4 share three subunits (ARP4, GOD1, and YAF9) and SWR1 and INO80 share four (ARP4, GOD1, RVB1, RVB2). This suggests that our algorithm is capable of identifying biologically relevant protein complexes, even those that share a significant number of subunits.

6 Conclusions and Future Work

In this paper, we study the clique cover problem on sparse networks as measured by treewidth and branchwidth, with an application to protein-complex discovery in protein-protein interaction networks. We give exact

polynomial-time algorithms for graphs with bounded treewidth and bounded branchwidth, and build on the latter using Baker's technique to obtain a polynomial time approximation scheme for planar graphs.

Our empirical studies show that the biological networks as well as synthetic networks with similar characteristics (e.g. edge density, degree distribution) indeed exhibit low treewidth, and our algorithms showed practical running times on various synthetic networks as well as biological networks. Moreover, our branchwidth based PTAS algorithm shows practical running time for computing solutions close to the optimal.

In proteomics research, existing experimental methods for detecting binary interactions often suffer from false negatives, i.e., some edges are not detected in the experiments. Therefore, in the direction towards hypergraph modelling of PPI networks, one may wish to cover the edges with quasi-cliques: Here, the optimization function needs to be modified slightly. One possible formulation may be to find a minimum cardinality quasi-clique cover, where a quasi-clique is defined by some lowerbound on the edge density. On the other hand, there may be classes of graphs other than the ones discussed here that admit polynomial time exact algorithms, for example, graphs with bounded genus or bounded degree.

References

[1] B. Adamcsek, G. Palla, I. J. Farkas, I. Derényi, and T. Vicsek. "CFinder: locating cliques and overlapping modules in biological networks". *Bioinformatics (Oxford, England)*, 22(8):1021–1023, April 2006.

[2] S. Arnborg, D. G. Corneil, and A. Proskurowski. "Complexity of finding embeddings in a k-tree". *SIAM Journal on Algebraic and Discrete Methods*, 8(2):277–284, April 1987.

[3] G. D. Bader and C. W. V. Hogue. "An automated method for finding molecular complexes in large protein interaction networks". *BMC bioinformatics*, 4:2, January 2003.

[4] B. Baker. "Approximation algorithms for NP-complete problems on planar graphs". *Journal of the ACM*, 41(1), 1994.

[5] A. L. Barabasi and R. Albert. "Emergence of scaling in random networks". *Science (New York, N.Y.)*, 286(5439):509–512, October 1999.

[6] H. L. Bodlaender. "A linear-time algorithm for finding tree-decompositions of small treewidth". *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[7] M. R. Cerioli, L. Faria, T. O. Ferreira, C. A. J. Martinhon, F. Protti, and B. Reed. "Partition into cliques for cubic graphs: planar case, complexity and approximation". *Discrete Applied Mathematics*, 156(12):2270–2278, 2008.

[8] M.-S. Chang and H. Müller. "On the tree-degree of graphs". In *Graph-theoretic concepts in computer science (Boltenhagen, 2001)*, 44–54, 2001.

[9] F. Chung, L. Lu, T. G. Dewey, and D. J. Galas. "Duplication Models for Biological Networks". *Journal of Computational Biology*, 10(5):677–687, October 2003.

[10] A. Denise and M. Vasconcellos. "The Random Planar Graph". *Congressus Numerantium*, 1996.

[11] F. Dorn, E. Penninkx, H. Bodlaender, and F. Fomin. "Efficient exact algorithms on planar graphs: Exploiting sphere cut branch decompositions". *Algorithms–ESA 2005*, 3669:95–106, 2005.

[12] F. V. Fomin and D. M. Thilikos. "New upper bounds on the decomposability of planar graphs". *Journal of Graph Theory*, 51(1):53–81, 2006.

[13] M. R. Garey and D. S. Johnson. *"Computers and Intractability : A Guide to the Theory of NP-Completeness"*. WH Freman, 1979.

[14] M. R. Garey, D. S. Johnson, and L. Stockmeyer. "Some simplified NP-complete graph problems". *Theoretical Computer Science*, 1(3):237–267, 1976.

[15] J. Gramm, J. Guo, F. Hüffner, and R. Niedermeier. "Data reduction and exact algorithms for clique cover". *ACM Journal of Experimental Algorithmics*, 13:2, 2009.

[16] Rudolf Halin. "s-functions for graphs". *Journal of Geometry*, 8:171–186, 1976. 10.1007/BF01917434.

[17] D. N. Hoover. "Complexity of graph covering problems for graphs of low degree". *Journal of Combinatorial Mathematics and Combinatorial Computing*, 11:187–208, 1992.

[18] H. B. Hunt III, M. V. Marathe, V. Radhakrishnan, and R. E. Stearns. "The complexity of planar counting problems". *SIAM Journal on Computing*, 27(4):1142–1167 (electronic), 1998.

[19] A. D. King, N. Przulj, and I. Jurisica. "Protein complex prediction via cost-based clustering". *Bioinformatics (Oxford, England)*, 20(17):3013–3020, November 2004.

[20] T. Kloks. *"Treewidth: Computations and Approximations (Lecture Notes in Computer Science)"*. Springer, 1 edition, September 1994.

[21] N. J. Krogan, G. Cagney, H. Yu, G. Zhong, X. Guo, A. Ignatchenko, J. Li, S. Pu, N. Datta, A. P. Tikuisis, T. Punna, J. M. Peregrín-Alvarez, M. Shales, X. Zhang, M. Davey, M. D. Robinson, A. Paccanaro, J. E. Bray, A. Sheung, B. Beattie, D. P. Richards, V. Canadien, A. Lalev, F. Mena, P. Wong, A. Starostine, M. M. Canete, J. Vlasblom, S. Wu, C. Orsi, S. R. Collins, S. Chandran, R. Haw, J. J. Rilstone, K. Gandi, N. J. Thompson, G. Musso, P. St Onge, S. Ghanny, M. H. Y. Lam, G. Butland, A. M. Altaf-Ul, S. Kanaya, A. Shilatifard, E. O'Shea, J. S. Weissman, C. J. Ingles, T. R. Hughes, J. Parkinson, M. Gerstein, S. J. Wodak, A. Emili, and J. F. Greenblatt. "Global landscape of protein complexes in the yeast Saccharomyces cerevisiae". *Nature*, 440(7084):637–643, March 2006.

[22] M. Li, J. Wang, J. Chen, Z. Cai, and G. Chen. "Identi-

fying the overlapping complexes in protein interaction networks". *International journal of data mining and bioinformatics*, 4(1):91–108, 2010.

[23] C. Lund and M. Yannakakis. "On the hardness of approximating minimization problems". *Journal of the ACM*, 41(5):960–981, 1994.

[24] E. Mujuni and F. Rosamond. "Parameterized complexity of the clique partition problem". *In Proc. Fourteenth Computing: The Australasian Theory Symposium (CATS 2008)*, 2008.

[25] J. Orlin. "Contentment in graph theory: covering graphs with cliques". *Indagationes Mathematicae (Proceedings)*, 80(5):406–424, 1977.

[26] N. Robertson and P. D. Seymour. "Graph minors. II. Algorithmic aspects of tree-width". *Journal of algorithms*, 7(3):309–322, 1986.

[27] N. Robertson and P. D. Seymour. "Graph minors. X. Obstructions to tree-decomposition". *Journal of Combinatorial Theory. Series B*, 52(2):153–190, 1991.

[28] V. Spirin and L. A. Mirny. "Protein complexes and functional modules in molecular networks". *Proceedings of the National Academy of Sciences of the United States of America*, 100(21):12123–12128, October 2003.

[29] H. Tamaki. *"A Linear Time Heuristic for the Branch-Decomposition of Planar Graphs"*, volume 2832:765–775 *ESA 2003 (Lecture Notes in Computer Science)*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[30] R. Uehara. "NP-complete problems on a 3-connected cubic planar graph and their application". *Technical report*, Tokyo Woman's Christian University, Tokyo, September 1996.