FPT Approximation for H-Hitting Set on Almost Chordal Graphs

Sri Krishna Sahoo
Department of Mathematics, Indian Institute of Technology Delhi, India

Ashutosh Rai
Department of Mathematics, Indian Institute of Technology Delhi, India

Saket Saurabh
The Institute Of Mathematical Sciences, HBNI, Chennai, India; University of Bergen, Bergen, Norway

Abstract

Given a graph $G = (V, E)$ and a fixed graph H of size k, we study H-Hitting Set where we need to find the smallest set $S \subseteq V$ such that $G \setminus S$ does not contain H as a subgraph. We also have that G is a modulated chordal graph with known modulator L of size l. Our result is a $O(\log(k^2 + kl))$-factor approximate solution that runs in FPT time with parameters k and l. For a chordal graph, we give an $O(\log k)$-factor approximation.

2012 ACM Subject Classification Replace ccsdesc macro with valid one

Keywords and phrases FPT algorithm, Approximation Algorithm, Modulated Chordal Graph

Digital Object Identifier 10.4230/LIPIcs.CVIT.2016.23

Funding *Sri Krishna Sahoo*: (Optional) author-specific funding acknowledgements
Ashutosh Rai: [funding]
Saket Saurabh: [funding]

1 Introduction

Let $\mathcal{F}$ be a class of graphs. In $\mathcal{F}$-Vertex Deletion problems, we are given a graph G and we must find the smallest set of vertices, S such that $G \setminus S$ belongs to $\mathcal{F}$. An example of such a problem is the Vertex Cover Problem where $\mathcal{F}$ is the set of all empty (edgeless) graphs and hence $G \setminus S$ must exclude all edges.

The problem has been studied when $\mathcal{F}$ is a class of graphs that have a bounded treewidth (Treewidth Modulator) or do not contain a path of length longer than k (k-Path Transversal), etc. In the problem in the focus of the current paper, we let H be a fixed graph and define $\mathcal{F}$ to be the class of graphs that exclude H as a subgraph. Following are the problems we solve.

H-Hitting Set on Chordal Graph

Input: Chordal graph $G = (V, E)$ and a fixed graph H of size k.
Output: Set $S \subseteq V$ such that $G \setminus S$ does not contain H as a subgraph.
Goal: Minimize |S|

H-Hitting Set on Modulated Chordal Graph

Input: Modulated chordal graph $G = (V, E)$ with known modulator, L of size l and a fixed graph H of size k.
Output: Set $S \subseteq V$ such that $G \setminus S$ does not contain H as a subgraph.
Goal: Minimize |S|

42nd Conference on Very Important Topics (CVIT 2016).
Editors: John Q. Open and Joan R. Access; Article No. 23; pp. 23:1–23:13
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1.1 Results

Our main results are the following.

▶ **Theorem 1.** *There is an FPT time $O(\log k)$-factor approximation algorithm for H-Hitting Set on a chordal graph.*

▶ **Theorem 2.** *There is an FPT time $O(\log(k^2 + kl))$-factor approximation algorithm for H-Hitting Set on a modulated chordal graph with modulator size l.*

Depending on the context, we assume G to be chordal when talking about the former problem, and modulated chordal when talking about the latter problem. For the chordal version of the problem, we shall use C to denote the solution with C^* being the optimal solution. For modulated chordal version of the problem, we shall use M to denote the solution with M^* denoting the optimal solution.

The rest of the paper is divided as follows. In preliminaries section we give necessary definitions and terminology. In techniques section, we have proved some basic lemmas and theorems to be used in our main work later. In prior work section, we have stated the theorems and results used from previous works that we use. In the section H-Hitting Set for Chordal Graph, we solve the simpler problem of finding the smallest H-Hitting Set given that the graph G is chordal giving a $\log(k)$-factor approximation in FPT time. This is followed by section H-Hitting Set for Modulated Chordal Graph where we prove the problem give that G is a modulated chordal graph with known modulator L. Finally, we have conclusion and references.

2 Preliminaries

2.1 Definitions

Clique: A complete graph is a simple undirected graph in which every pair of distinct vertices is connected by a unique edge. A clique of a graph G is an induced subgraph of G that is complete. A maximum clique of a graph, G, is a clique, such that there is no clique with more vertices.

Excluding the graph H from graph $G = (V, E)$ means removing a subset $S \subseteq V$ from V so that there does not exist a subset of vertices $X \subseteq V(G)$ and edges $Y \subseteq E(G)$ such that the subgraph $G' = (X, Y)$ is isomorphic to H. We sometimes use the term "exclude an instance" of H from G when we remove one subset $H' \subseteq V(G)$ from G which is isomorphic to H. This might result in a graph G' that still has subgraphs isomorphic to H. It should be clear from the context, the definition of "exclusion".

Tree Decomposition[5]: A Tree decomposition of graph G is a pair $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$, where T is a tree whose every node t is assigned a vertex subset $X_t \subseteq V(G)$, called a bag, such that the following three conditions hold:

1. $\bigcup_{t \in V(T)} X_t = V(G)$. In other words, every vertex of G is in at least one bag.
2. For every $uv \in E(G)$, there exists a node t of T such that bag X_t contains both u and v.
3. For every $u \in V(G)$, the set $T_u = t \in V(T) : u \in X_t$, i.e., the set of nodes whose corresponding bags contain u, induces a connected subtree of T.

Treewidth[5]: Width of tree decomposition $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ equals $\max_{t \in V(T)} |X_t| - 1$, that is, the maximum size of its bag minus 1. The treewidth of a graph G, denoted by $\mathrm{tw}(G)$, is the minimum possible width of a tree decomposition of G.

Chordal Graph: A graph $G = (V(G), E(G))$ is called a chordal graph when it is a simple graph in which every graph cycle of length four and greater has a cycle chord. In other words, a chordal graph is a graph possessing no chordless cycles of length four or greater[3,4]. Equivalently, a graph G is chordal if and only if it has a tree decomposition such that every bag induces a clique. The latter definition would be used in our analysis.

Modulated Chordal Graph: A graph, $G = (V, E)$ is called a modulated chordal graph when its vertex set V can be written as a disjoint union of two sets $C \subseteq V$ and $L \subseteq V$ such that the subgraph induced by C is chordal. L is called the modulator. We will use the terms "chordal part of G" to denote the vertices, $v \in C$ and "modulator part of G" for vertices $v \in L$.

We define the q-Subset Vertex Separator (q-SVS) problem which has a bicriteria approximation algorithm stated in Prior Results Section.

q-SVS: Given graph $G = (V, E)$, a subset $R \subseteq V$ and an integer q, we need to find the smallest set $S \subseteq V$ such that any connected component in $G \setminus S$ has at most q vertices from R. We can define such an instance of the problem by the ordered triplet (G, R, q).

We now state a basic lemma to be used in further analysis proved by Gupta et al.[2] (as Lemma 10).

▶ **Lemma 3.** *Suppose graph G has its treewidth bounded by $t - 1$ and let $P \subseteq V(G)$. Then for each natural number δ there exists a set $W \subseteq V(G)$ such that $|W| \leq \frac{t}{\delta} \cdot |P|$ and each connected component of $G \setminus W$ contains at most δ elements from P. What is more, if the tree decomposition is given, the set W can be constructed in polynomial time.*

We will later use the lemma on the instance (G, t, P, δ).

2.2 Monadic Second-Order Logic on Graphs

Here we describe the basic terminology and definitions used in Monadic Second-Order Logic. The reader is referred to the book Parameterised Algorithms by Cygan et al.[8] for a more detailed introduction to the topic.

MSO_2 is a formal language of expressing properties of graphs and objects inside these graphs, such as vertices, edges, or subsets of them. A formula φ of **MSO_2** consists of -

1. *Variables* are of four types: Single vertices, single edges, subsets of vertices, and subsets of edges.
2. *Free variables* are variables that are given from "outside", whose properties we verify in the graph.
3. *Boolean operators* such as such as $\neg$ (negation, logical NOT), $\wedge$ (conjunction, logical AND), $\vee$ (disjunction, logical OR), and $\implies$ (implication).
4. Quantifiers such as $\forall$ (forall) and $\exists$ (exists).

Such a formula φ evaluates to *true* or *false* based on the graph G it is applied to. We use the time complexity guarantee for such an evaluation by the **Courcelle's theorem[6]**.

▶ **Theorem 4.** ***Courcelle's Theorem.*** *Assume that φ is a formula of **MSO_2** and G is an n-vertex graph equipped with evaluation of all the free variables of φ. Suppose, moreover, that a tree decomposition of G of width t is provided. Then there exists an algorithm that verifies whether φ is satisfied in G in time $f(||\varphi||, t) \cdot n$, for some computable function f.*

2.3 Self-Reducibility and Search vs. Decision

Courcelle's Theorem in the previous section deals with the decision problem of asking *whether* a solution exists. But our aim here is to solve the SUBGRAPH ISOMORPHISM problem where we want to find a subgraph, $S \subseteq G$ such that it is isomorphic to a given graph H. In cases where the treewidth of G is bounded, we apply the Courcelle's Theorem to decide whether such a subgraph exists. However, we want to find that subset to construct out approximate solution.

We lay out the algorithm to find the solution of such a search problem given we have an algorithm to its decision counterpart without going into the detail of the NP-hardness theory behind it[10]. Say, $\phi(v_1, v_2, ..., v_n)$ is a formula on n boolean variables, $v_1, v_2, ..., v_n$ which represents the solution of the decision problem of whether G has a subgraph isomorphic to H. Here v_i represents the i^{th} vertex of graph, G and can be assigned the values 0 or 1 which means excluding or including the vertex in the decision problem respectively. Specifically, if $b_1, b_2, ..., b_l \in \{0, 1\}(l \leq n)$, then by $\phi|_{b_1,...b_l}$ we mean the formula on the variables $v_{l+1}, ..., v_n$ obtained by setting $v_1 = b_1 ..., v_l = b_l$ in ϕ, i.e. $\phi(b_1, b_2, ..., b_l, v_{l+1}, ..., v_n)$. Let us call IS-POSSIBLE($\phi(v_1, v_2, ..., v_n)$) as the solution of decision problem. Algorithm 1 gives the solution to the corresponding search problem in the form of an assignment of the vertices.

Algorithm 1 FPT algorithm for SUBGRAPH ISOMORPHISM problem

1: Initialize $a_i \forall i \in \{1, 2, ..., n\}$
2: **for** $i \leftarrow 1$ to n **do**
3: $\quad \phi' \leftarrow \phi(v_i = 0)$
4: $\quad \phi'' \leftarrow \phi(v_i = 1)$
5: $\quad$ **if** IS-POSSIBLE(ϕ') **then**
6: $\quad\quad a_i \leftarrow 0$
7: $\quad\quad \phi \leftarrow \phi(a_0, ..., a_{i-1}, 0, v_{i+1}, ...v_n)$
8: $\quad$ **else**
9: $\quad\quad$ **if** IS-POSSIBLE(ϕ'') **then**
10: $\quad\quad\quad a_i \leftarrow 1$
11: $\quad\quad\quad \phi \leftarrow \phi(a_0, ..., a_{i-1}, 1, v_{i+1}, ...v_n)$
12: $\quad\quad$ **else**
13: $\quad\quad\quad$ **return** IMPOSSIBLE
14: $\quad\quad$ **end if**
15: $\quad$ **end if**
16: **end for**
17: **return** $(a_1, a_2, ..., a_n)$

Proof of Correctness: We now prove that Algorithm 1 gives a correct solution of the search problem. The solution contains a subgraph isomorphic to H since at $i = n$, either IS-POSSIBLE(ϕ') or IS-POSSIBLE(ϕ'') is TRUE. Next, we prove that the solution has exactly k vertices. Observe that if it contains $k + j$ vertices, there must be vertices $\{v_{r_i}\}, r_i < n, i \in \{1, 2, ..., j\}$ in the solution which even when removed, would render the solution feasible. This contradicts line 5 of the algorithm as in this case, v_{r_i} would not have been picked up in this case in the first place. Now, since the solution has exactly k vertices and has a subgraph isomorphic to H (of size k), the solution must itself be the subgraph.

3 Techniques

In this section, we discuss some preliminary results to be used later sections.

3.1 Largest Clique in a Chordal Graph

In this subsection, we provide a brief outline of a poly-time algorithm to find the largest clique in a chordal graph.

We have a linear time algorithm to find a perfect elimination ordering for any chordal graph[7]. To find the largest (maximum) clique in a graph G, we can first find all the cliques in the graph and then find the largest. To list all cliques, find a perfect elimination ordering and form a clique for each vertex v with its neighbors that are later than v in the perfect elimination ordering.

3.2 Factor k-approximation for H-Hitting Set on Chordal Graph

In this subsection we state a result that originates from Courcelle's Theorem[6]. We give an FPT time k-approximation algorithm for the H-Hitting Set on a chordal graph. that we will later use as a basis to find the $\log(k)$-factor approximation.

▶ **Theorem 5.** *Given a chordal graph G and a fixed graph H of size k, there is a k-approximation algorithm for H-Hitting Set problem on G that runs in FPT-time with parameter k.*

We give a constructive algorithm to build the approximate solution, P. To find P on a graph $G = (V, E)$, we keep finding subsets $S \subseteq V$ such that a subgraph on the vertex set S is isomorphic to H and do

(i) $V := V \setminus S$
(ii) $P := P \cup S$

until we cannot find any more such S. We call this "excluding" H from G. We first claim that this results in an P that is a k-approximate solution, then we describe the algorithm to find such a subset S.

▷ Claim 6. The resulting set P is a k-approximate solution for k-Hitting Set

Proof. We first prove that P is a feasible solution for the problem and prove the bound on its size to prove the approximation claim.

The first part is trivial since by definition we are removing every subset $S \subseteq V$ such that S is isomorphic to H, hence G would not contain a subgraph isomorphic to H. For the second part, observe that $|H| = k$, and let P^* be the optimal solution. Observe that since P^* is the optimal (minimal) solution, removing any vertex from P^* would result in G having an occurrence of an S isomorphic to H. But by the construction of P, we must have removed such an S by adding at most k vertices to P. Hence for every vertex in P^*, we have at most k vertices in P proving the approximation factor. ◀

Now we describe the algorithm for finding all such instances S and removing them. We do this in a two step process based on the treewidth of G-

Step 1 Excluding H from G when treewidth of $G \geq k$.

Step 2 Excluding H from the graph G' (which has a bounded treewidth) completely. In this step we completely exclude

For step 1, we prove that Algorithm 2 returns a subset $P \subseteq V(G)$

▶ **Theorem 7.** *Algorithm 2 only removes subgraphs isomorphic to H from G and returns the union of their vertex sets as P. Algorithm 2 also renders the treewidth, $tw(G) < k$.*

Proof. We will first prove that the treewidth of the residual graph G after running the algorithm is less than k. Then we prove that we are only excluding instances of H from G, and hence are respecting the approximation factor of the solution.

Since chordal graphs have the hereditary property, the subgraph induced on $G \backslash H_C \subseteq G$ is chordal as well. As $G \backslash H_C$ is chordal, there exists a tree decomposition, $\mathcal{T}' = (T', \{X_t\}_{t \in V(T')})$ of graph G in which all bags, $X_t, t \in V(T')$ are cliques. Looking at the condition of the *while* loop, we see that the algorithm ends when the size of the largest clique in G is smaller than k. This implies that the largest bag, X_{t^*} in $\mathcal{T}'$ has size less than k. In other words, when the algorithm ends,

$$\max_{t \in V(T')} |X_t| - 1 < k - 1 \implies \text{tw}(G) < k$$

◀

Algorithm 2 Bounding the treewidth of G

```
1: Initialize set P                              ▷ To be returned by the algorithm
2: Initialize C ← largest clique in G
3: while |C| ≥ k do
4:     H_C ← pick any set of k vertices from C   ▷ Any k vertex subset is isomorphic to H
5:     G ← G \ H_C
6:     C ← largest clique in G
7: end while
8: return P
```

For step 2, We define the existence of a subgraph of G isomorphic to H by monadic second-order logic on graphs. By *Courcelle's Theorem*, this can be evaluated in FPT-time with the parameters treewidth t and length of the encoding of the MSO2 formula, $||\varphi||$. By section 2.3, we have that given an FPT algorithm to solve the decision problem, we have an FPT algorithm to solve the corresponding search problem as well.

We are given graphs $G = (V(G), E(G))$ and $H = V(H), E(H))$. Say, $V(H) = \{v_1, v_2, ..., v_k\}$. Let us represent the edge between vertices v_i and v_j as $e(v_i, v_j) \in E(H)$. The following formula represents the condition of existence of a subgraph of G isomorphic to H.

$$\varphi(G, H) = \exists v_1', v_2', ..., v_k' \in V(G) \ni \forall e = (v_i, v_j) \in E(H) \exists e' = (v_i', v_j') \in E(G)$$

3.3 k-approximation for H-Hitting Set on Modulated Chordal Graph

▶ **Theorem 8.** *There is a k-approximation algorithm for H-HITTING SET on modulated chordal graphs that runs in FPT-time with parameter k.*

Proof. Following the notations from preliminaries, let G be a modulated chordal graph with C and L being its chordal and modulator parts respectively. Similar to section 3.2, we find a factor k-approximation by completely "excluding" H from G. Here, however, we can not use the same two steps, as G is not chordal and hence we can no longer find the largest clique in polynomial time. Despite this, we follow the same principle of looking for subgraphs of G isomorphic to H and collecting them to give a factor k-approximate solution

As a sub-problem, we solve the Subgraph Isomorphism problem, where we find a subset $S \subseteq V(G)$ such that a subgraph on S is isomorphic to H in order to remove them. We have the following three cases -

1. $S \subseteq C$
2. $S \subseteq L$
3. $S \cap C \neq \phi$ and $S \cap L \neq \phi$

We describe the method to solve each of the above in the following ways -

1. This case is the same as Section 3.2 since C is chordal. We use the same two step process of bounding treewidth by finding the largest clique (Algorithm 2) followed by using MSO_2 logic on bounded treewidth graph.
2. Consider the vertices of H in an order $\{h_1, h_2, ..., h_k\}$. Select a set of k vertices, $S \subseteq V(L)$ in $\binom{l}{k}$ ways. Consider a permutation of the k vertices as $S = \{v_1, v_2, ..., v_k\}$. For every edge between vertices, h_i and h_j of H, check whether the corresponding vertices, v_i and v_j in K have an edge between them, i.e., whether $\exists e' = (v_i, v_j) \in E(L) \forall e = (h_i, h_j) \in E(H)$ is true. Check this for every combination of k vertices from L and every permutation of those k vertices. If the statement is true for any of permutation of any subset, S^* of k vertices, we return S^* as an isomorphic subgraph. Time complexity for this algorithm is $\binom{l}{k} k! |E(H)|$ and hence FPT in parameters l and k.
3. Here, we first guess the set $S_L = S \cap L$ by first guessing the size $k_l = |S_L|$ followed by guessing the k_l vertices from L in $\binom{l}{k_l}$ ways. After excluding H from C in case 1, call the remaining graph C_H. Now excluding H from G is restricted to excluding it from $S_L \cup C_H$. We do the latter by constructing a tree decomposition of $S_L \cup C_H$ with bounded treewidth and using MSO_2 logic similar to Section 3.2.

   Note that $\mathrm{tw}(C_H) < k$ (Theorem 7). Consider a tree decomposition, $\mathcal{T} = (T, \{X_t\}_{t \in V(T)})$ of C_H. We construct a tree decomposition of $S_L \cup C_H$, $\mathcal{T}' = (T, \{X_t \cup S_L\}_{t \in V(T)})$ and note that $\mathrm{tw}(S_L \cup C_H) < k + k_l \leq 2k$. Then use Courcelle's theorem similar to Section 3.2 on the graph with bounded treewidth.

◀

3.4 FPT algorithm on optimal solution size for H-Hitting Set

In this section, we describe an FPT time, with parameters k and l, exact algorithm to solve H-Hitting Set on a general graph G. The algorithm is FPT on the parameter optimal solution size, OPT and $|H| = k$.

This branching algorithm has two major parts- Subgraph Isomorphism problem, and given such a subgraph H' isomorphic to H, branching on each of its vertices and recursively solving the problem. In the branching step, we remove each vertex $v \in V(H')$ one at a time from V and such that G' is the graph induced by the vertex set $V \setminus \{v\}$. Then recursively look for subgraphs of G' isomorphic to H. Once we cannot find any such subgraph, we have completely excluded H from G. All the vertices we have deleted from V to reach this subgraph together make up a solution for the H-Hitting Set problem on the graph G. An optimal solution is one where we have to remove least such vertices.

Similar to the branching algorithm for finding an optimal solution of Vertex Cover problem in FPT time with parameter as the optimal solution size, here we form a branching tree, T where every node is an induced subgraph of G after removing some vertices from it. Root of this tree is G itself. We traverse this tree in a level-by-level manner, where the depth of a level denotes the number of vertices removed from G. The first time (least depth) we find a subgraph $G' \subseteq G$ in which we have completely excluded H, we can collect the set of vertices removed at every level and return that as an optimal solution. Algorithm 3 describes the algorithm in the form of a pseudo-code.

Time complexity: Given the size of the optimal solution, OPT, one can stop branching at a node whose depth is OPT as we cannot find an optimal solution from this node. Hence we limit the maximum depth of the tree to OPT. At every node in the tree, we have $k = |H|$ branches. This gives the total number of nodes in the tree, $|T| \leq k^{OPT}$. At every node, we search for a subgraph isomorphic to H which takes FPT time with parameter k (Section 2.3). Together, we have an FPT time complexity with parameters k and OPT.

Algorithm: We use a Breadth First Search type algorithm to construct the tree using a **queue** data structure (First In First out). **queue** supports three operations **queue.push()** and **queue.get()** to put and remove objects from the **queue**, and **queue.hasElements()** that returns whether the queue is not empty.

Algorithm 3 FPT in optimal solution size algorithm for H-Hitting Set

```
Initialise G                                   ▷ Induced subgraph after removing vertices
Initialize l ← OPT                             ▷ Maximum depth of branching tree
Initialize S ← ϕ                               ▷ Collection of vertices to create an Optimal Solution
Initialize queue = ()                          ▷ To implement breadth-first search
queue.push(G, l, S)
while queue.hasElements() do
    (G, l, S) = queue.get()                    ▷ Get instance in FIFO manner
    Solve Subgraph Isomorphism on G and find H'
    if found then
        if l = 0 then
            Solution not found                 ▷ Solution size exceeded OPT
        else
            for v ∈ V(H') do
                G ← G \ v
                l ← l − 1
                S ← S ∪ v
                queue.push(G, l, S)
            end for
        end if
    else
        Return S                               ▷ S is an optimal solution
    end if
end while
```

4 Prior Results

4.1 q-Subset Vertex Separator

We now state the following result proved by Lee[1] which we will relate with the H-Hitting Set problem and use its result for an approximation of the latter.

▶ **Theorem 9.** *For any $\epsilon \in (0, 1/4]$, there is a $\left(\frac{1}{1-2\epsilon}, O(\frac{\log q}{\epsilon})\right)$-bicriteria approximation algorithm for q-Subset Vertex Separator problem that runs in time $n^{O(1)}$.*

In other words, if the optimal solution for the problem is S_q^*, the algorithm returns an approximate solution $\tilde{S}_q$ such that $|\tilde{S}_q| \leq |S_q^*| \cdot O(\frac{\log q}{\epsilon})$ and every connected component C in $G \setminus \tilde{S}$ satisfies $|C \cap R| \leq \frac{q}{1-2\epsilon}$. As a direct application, we can suitably substitute $\epsilon = 1/4$ and get a factor $O(\log q)$ approximate solution and connected components having at most $2q$ vertices from R.

5 H-Hitting Set for Chordal Graphs

We now outline the algorithm to find a factor $O(\log k)$-approximation for the H-Hitting Set problem on a chordal graph G. In the next section about the same problem for modulated chordal graphs, we will use the same ideas to a large extent. The algorithm has three main steps.

We are going to relate two different problems here so let us redefine the notations used for each. Let $G = (V, E)$ be a chordal graph, H be a fixed graph of size k. Let the optimal solution for the H-Hitting Set problem on G is C^*. Let us define an instance of the q-SVS problem on the triplet $(G, R = C', q = 2k^2)$. Let its optimal solution is S_q^*

Step 1 Find a factor $f(k)$-approximate solution, C' for the H-Hitting Set problem.

Step 2 Find a bound on the treewidth of $G \setminus C^*$ to use Lemma 3. Then relate the optimal solutions for H-Hitting Set with an instance of the q-SVS problem. Show that an approximate solution, $\tilde{S}_q$ of q-SVS problem is within an $O(\log k)$ factor approximation for C^*.

Step 3 Run an exact algorithm in FPT time (on optimal solution size) to solve H-Hitting Set on the set $G \setminus \tilde{S}_q$. Combine $\tilde{S}_q$ and the above exact solution and return as the final factor $O(\log k)$ solution of H-Hitting Set.

As the first step, C' is a factor k-approximate solution(Section 3.2)

For the second step, we first give a bound on the treewidth of $G \setminus \tilde{S}_q$ and then relate the optimal solutions of H-Hitting Set problem with q-SVS problem on the instance $(G, R = C', q = k^2)$.

▷ Claim 10. $\mathrm{tw}(G \setminus C^*) < k$

Proof. We prove this by contradiction. Assume $\mathrm{tw}(G \setminus C^*) \geq k$, then there exists a bag with at least k vertices in every tree decomposition, $\mathcal{T}$ of $G \setminus C^*$. Since $G \setminus C^*$ is an induced subgraph of a chordal graph, it is chordal as well. Hence, $G \setminus C^*$ has a tree decomposition, $\mathcal{T}^* = (T, \{X_t\}_{t \in V(T)})$ in which every bag, X_t is a clique. Since the largest bag, X_{t^*} in $\mathcal{T}^*$ is larger than $\mathrm{tw}(G \setminus C^*) + 1$, it contains at least k vertices. Since X_{t^*} is a complete graph of size at least k, H is its subgraph. Hence contradicting the feasibility of C^* as a solution. ◀

▷ Claim 11. $|S_q^*| \leq 2|C^*|$

Proof. Observe that $(G \setminus C^*, t = k, P = C', \delta = k^2)$ is a feasible instance to apply lemma 3. Using Lemma 3 on $G \setminus C^*$ with $\delta = k^2$, we obtain a set $|W| \leq \frac{k|C'|}{\delta}$ such that every connected component in $G \setminus (S^* \cup W)$ has at most k^2 vertices from C'. This makes $C^* \cup W$ a feasible solution of the instance, $(G, R = C', q = k^2)$ of q-SVS problem which means $|S_q^*| \leq |C^* \cup W|$. Substituting the above bound for $|W|$, and the fact that C' is a k-factor approximation for H-Hitting Set problem, we obtain the result.

$$|W| \leq \frac{k|C'|}{\delta} \leq \frac{k \cdot k|C^*|}{k^2} \leq |C^*|$$

Substituting in $|S_q^*| \leq |C^* \cup W| \leq |C^*| + |W|$, we get $|S_q^*| \leq 2|C^*|$. ◀

From Theorem 9, $|\tilde{S}_q| \leq O(4 \log(k^2))|S_q^*|$. Using properties of big-O, logarithms and claim 10, we have

$$|\tilde{S}_q| \leq O(\log k)|C^*| \tag{1}$$

We now have a set $\tilde{S}_q$ which is within a factor $O(\log k)$-approximation for H-Hitting Set problem. But it is not feasible since we have not excluded H from $G \setminus \tilde{S}_q$.

As the third step, We now undertake the exclusion part of the problem and claim that we can exclude H from $G \setminus \tilde{S}_q$ in FPT time (with parameter k). We will then prove that the exact solution, X^* of the H-Hitting Set solved on the graph $G \setminus \tilde{S}_q$ can be combined with $\tilde{S}_q$ and we have that $\tilde{C} = X^* \cup \tilde{S}_q$ is a factor $O(\log k)$-approximate solution for H-Hitting Set on the graph G.

▷ Claim 12. There is an FPT time exact algorithm to exclude H from $G \setminus \tilde{S}_q$.

Proof. Let the connected components in $G \setminus \tilde{S}_q$ are $\{C_i\}, i = 1, 2, ..., m$. From Theorem 9, we have that every connected component, C_i in $G \setminus \tilde{S}_q$ has at most k^2 vertices from C'.

Say $C_i' = C_i \cap C'$. C_i' is a feasible solution of H-Hitting Set on C_i (Remember that C' excludes H from the whole graph and hence from each component as well). This gives a bound on the optimal solution X_i^* of H-Hitting Set problem on the subgraph C_i $(= |C_i'|$. From Section 3.4, we have an FPT time algorithm, with parameter as the optimal size of the solution, for H-Hitting Set. Since the bound on optimal solution size is $|C_i'| = k^2$, we have an FPT time algorithm with parameter k proving the claim. ◀

Let $X^* = \cup_{i=1}^m X_i^*$ and $\tilde{C} = X^* \cup \tilde{S}_q$. Now, we prove that $\tilde{C}$ is an $O(\log k)$-approximate solution to H=Hitting Set on the chordal graph G.

▶ **Theorem 13.** *$\tilde{C}$ is a factor $O(\log k)$-approximate solution for H-Hitting Set on the chordal graph G.*

Proof. This solution is feasible since we have excluded H from $G \setminus \tilde{S}_q$ by the definition of X^*. For the size of the approximate solution, we have

$$|\tilde{C}| \leq |X^*| + |\tilde{S}_q| \leq |C^*| + O(\log k)|C^*| \leq O(\log k)|C^*|$$

We have used equation (1) for the above inequality. To prove that $|X^*| \leq |C^*|$, say $C_i^* = C^* \cap C_i$, i.e. vertices of the optimal solution that belong to the component C_i. See that C_i^* is a feasible solution of H-Hitting Set on C_i and hence its optimal solution, X_i^* satisfies $|X_i^*| \leq |C_i^*|$. Now, $|X^*| \leq \sum_{i=1}^m |X_i^*| \leq \sum_{i=1}^m |C_i^*| \leq |C^*|$. ◀

Hence we have $\tilde{C}$, a factor $O(\log k)$-approximate solution for H-Hitting Set on chordal graph, G proving Theorem 1.

6 H-Hitting Set for Modulated Chordal Graphs

In this section, we shall take on proving Theorem 2 and giving an $O(\log(k^2 + kl))$-approximation algorithm for H-Hitting Set problem on modulated chordal graph G with given modulator L of size l. Again, H is a fixed graph of size k. Following the notation standards, let M^*, M' and $\tilde{M}$ be the optimal solution, factor k, and factor $O(\log(k^2 + kl))$ approximate solutions for said problem. Here, we talk about q-SVS problem on the instance $(G, R = M', q = k(k + l))$ $(\epsilon = \frac{1}{4})$ whose optimal solution is again denoted by S_q^* and approximate solution (from theorem 9) by $\tilde{S}_q$. Note that the same symbols are used here for the solutions but they are solutions for separate instances of the problem. Like the previous section, we will use the same three main steps to give the result.

As a first step, we already have a factor k-approximate solution, M' from Section 3.3.

For the second step, we first give a treewidth bound and then relate the optimal solutions of H-Hitting Set problem on the modulated chordal graph.

▷ Claim 14. $\mathrm{tw}(G \setminus M^*) < k + l$

Proof. We will construct a tree decomposition, $\mathcal{T}_G = (T_G, \{A_t\}_{t\in V(T_G)})$ with width less than $k + l$ of $G \setminus M^* = (V, E)$. Considering the chordal and modulator parts of G, consider a tree decomposition $\mathcal{T}_C = (T_C, \{B_t\}_{t\in V(T_C)})$ of the induced subgraph on $C \setminus M^* \subseteq V$. Construct $\mathcal{T}_G$ as follows:

1. $T_G = T_C$, and
2. $A_t = B_t \cup L \forall t \in V(T_G) = V(T_C)$

One can check that all the properties of a tree decomposition are satisfied by the above construction of $\mathcal{T}_G$.

To prove the treewidth bound, observe that in $\mathcal{T}_G$, the size of every bag, $|A_t| = |B_t \cup L|$. Since $B_t \subseteq C$ which is disjoint from L, we have $|A_t| = |B_t| + |L|$. From Claim 10, $\mathrm{tw}(C \setminus M^*) < k \implies |B_t| \leq k$. Hence we have $|A_t| \leq l + k \implies \mathrm{tw}(G \setminus M^*) < k + l$ ◀

▷ Claim 15. $|S_q^*| \leq 2|M^*|$

Proof. Observe that $(G \setminus M^*, t = k + l, P = M', \delta = k^2 + kl)$ is a feasible instance to apply lemma 3. Using Lemma 3 on this instance, we obtain a set $|W| \leq \frac{t|M'|}{\delta}$ such that every connected component in $G \setminus (M^* \cup W)$ has at most $k^2 + kl$ vertices from M'. This makes $M^* \cup W$ a feasible solution of the instance, $(G, R = M', q = k^2 + kl)$ of q-SVS problem which means $|S_q^*| \leq |M^* \cup W|$. Substituting the above bound for $|W|$, and the fact that M' is a k-factor approximation for H-Hitting Set problem, we obtain the result.

$$|W| \leq \frac{(k + l)|M'|}{\delta} \leq \frac{(k + l) \cdot k|M^*|}{k(k + l)} \leq |C^*|$$

Substituting in $|S_q^*| \leq |M^* \cup W| \leq |M^*| + |W|$, we get $|S_q^*| \leq 2|M^*|$. ◀

From Theorem 9, $|\tilde{S}_q| \leq O(4 \log(k^2 + kl))|S_q^*|$. Using properties of big-O, logarithms and claim 15, we have

$$|\tilde{S}_q| \leq O(\log(k^2 + kl))|C^*| \tag{2}$$

Similar to previous section, we have an infeasible factor-$O(\log k^2 + kl)$ approximation for H-Hitting Set problem. We will again exclude H from $G \setminus \tilde{S}_q$ in FPT time (with parameters k, and l)We will then prove that the exact solution, Y^* of the H-Hitting Set solved on the graph $G \setminus \tilde{S}_q$ can be combined with $\tilde{S}_q$ and we have that $\tilde{M} = Y^* \cup \tilde{S}_q$ is a factor $O(\log(k^2 + kl))$-approximate solution for H-Hitting Set on the modulated chordal graph G.

▷ Claim 16. There is an FPT time exact algorithm to exclude H from $G \setminus \tilde{S}_q$.

Proof. Let the connected components in $G \setminus \tilde{S}_q$ are $\{C_i\}, i = 1, 2, ..., m$. Again, from Theorem 9, we have that every connected component, C_i in $G \setminus \tilde{S}_q$ has at most $q = k^2 + kl$ vertices from C'. Again, with Algorithm 3, we optimally exclude H from each such connected component in FPT time with parameter k and l. Since the number of components is linearly bounded by n $(= |G|)$. We have an FPT time algorithm for excluding H from $G \setminus \tilde{S}_q$. ◀

Let $Y^* = \cup_{i=1}^m Y_i^*$ and $\tilde{M} = Y^* \cup \tilde{S}_q$. Now, we prove the approximation factor guarantee.

▶ **Theorem 17.** *$\tilde{M}$ is a factor $O(\log k^2 + kl)$-approximate solution for H-Hitting Set on the modulated chordal graph G.*

Proof. This solution is feasible since we have excluded H from $G \setminus \tilde{S}_q$. For the size of the approximate solution, using the same arguments as Theorem 12, we have

$$|\tilde{M}| \leq |Y^*| + |\tilde{S}_q| \leq |M^*| + O(\log(k^2 + kl))|M^*| \leq O(\log(k^2 + kl))|M^*|$$

◀

Hence we have $\tilde{M}$, a factor $O(\log(k^2 + kl))$-approximate solution for H-Hitting Set on modulated chordal graph, G proving Theorem 2.

7 Conclusion

Morbi eros magna, vestibulum non posuere non, porta eu quam. Maecenas vitae orci risus, eget imperdiet mauris. Donec massa mauris, pellentesque vel lobortis eu, molestie ac turpis. Sed condimentum convallis dolor, a dignissim est ultrices eu. Donec consectetur volutpat eros, et ornare dui ultricies id. Vivamus eu augue eget dolor euismod ultrices et sit amet nisi. Vivamus malesuada leo ac leo ullamcorper tempor. Donec justo mi, tempor vitae aliquet non, faucibus eu lacus. Donec dictum gravida neque, non porta turpis imperdiet eget. Curabitur quis euismod ligula.

8 References

1. Lee E. Partitioning a Graph into Small Pieces with Applications to Path Transversal https://arxiv.org/abs/1607.05122
2. tw-t modulator paper
3. West, D. B. "Chordal Graphs" and "Chordal Graphs Revisited." Introduction to Graph Theory, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, pp. 224-226 and 323-328, 2000.
4. Gross, J. T. and Yellen, J. Graph Theory and Its Applications, 2nd ed. Boca Raton, FL: CRC Press, 2006.
5. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. 2015. Parameterized Algorithms (1st. ed.). Springer Publishing Company, Incorporated.
6. Courcelle, B.: The monadic second-order logic of graphs I: Recognizable sets of finitegraphs. Information and Computation 85, 12-75 (1990)

7. D.J. Rose, R.E. Tarjan, G.S. Lueker, Algorithmic aspects of vertex elimination on graphs, SIAM J. Comput. 5 (2) (1976) 266–283. MR 408312.
8. Parameterixed algorithms book by cygan et al.
9. color coding by noga alon paper
10. S. Arora and B. Barak. Computational Complexity: A Modern Approach. Cambridge University Press, 2009.