ALGORITHMIC LOWER BOUNDS FOR PROBLEMS PARAMETERIZED BY CLIQUE-WIDTH

Fedor V. Fomin* Petr A. Golovach Daniel Lokshtanov
Saket Saurabh
Department of Informatics, University of Bergen, N-5020 Bergen, Norway
{fedor.fomin|petr.golovach|daniello|saket.saurabh}@ii.uib.no

Abstract

Many NP-hard problems can be solved efficiently when the input is restricted to graphs of bounded tree-width or clique-width. In particular, by the celebrated result of Courcelle, every decision problem expressible in monadic second order logic is fixed parameter tractable when parameterized by the tree-width of the input graph. On the other hand if we restrict ourselves to graphs of clique-width at most t, then there are many natural problems for which the running time of the best known algorithms is of the form $n^{f(t)}$, where n is the input length and f is some function. It was an open question whether natural problems like GRAPH COLORING, MAX-CUT, EDGE DOMINATING SET, and HAMILTONIAN PATH are fixed parameter tractable when parameterized by the clique-width of the input graph. As a first step toward obtaining lower bounds for clique-width parameterizations, in [*SODA 2009*], we showed that unless FPT$\neq$W[1], there is no algorithm with run time $O(g(t) \cdot n^c)$, for some function g and a constant c not depending on t, for GRAPH COLORING, EDGE DOMINATING SET and HAMILTONIAN PATH. But the lower bounds obtained in [*SODA 2009*] are weak when compared to the upper bounds on the time complexity of the known algorithms for these problems when parameterized by the clique-width.

In this paper, we obtain the asymptotically tight bounds for MAX-CUT and EDGE DOMINATING SET by showing that both problems

- cannot be solved in time $f(t)n^{o(t)}$, unless Exponential Time Hypothesis (ETH) collapses; and
- can be solved in time $n^{O(t)}$,

where f is an arbitrary function of t, on input of size n and clique-width at most t.

We obtain our lower bounds by giving non-trivial structure-preserving "linear FPT reductions".

*Partially supported by the Norwegian Research Council

1 Introduction

Tree-width is one of the most fundamental parameters in Graph Algorithms. Graphs of bounded tree-width enjoy good algorithmic properties similar to trees and this is why many problems which are hard on general graphs can be solved efficiently when the input is restricted to graphs of bounded tree-width. On the other hand, many hard problems also become tractable when restricted to graphs "similar to complete graphs". Courcelle and Olariu [6] introduced the notion of clique-width which captures nice algorithmic properties of both extremes.

Since 2000, the research on algorithmic and structural aspects of clique-width is an active direction in Graph Algorithms, Logic, and Complexity. Corneil, Habib, Lanlignel, Reed, and Rotics [4] show that graphs of clique-width at most 3 can be recognized in polynomial time. Fellows, Rosamond, Rotics, and Szeider [11] settled a long standing open problem by showing that computing clique-width is NP-hard. Oum and Seymour [27] describe an algorithm that, for any fixed t, runs in time $O(|V(G)|^9 \log |V(G)|)$ and computes $(2^{3t+2}-1)$-expressions for a graph G of clique-width at most t. Recently, Hliněný and Oum obtained an algorithm running in time $O(|V(G)|^3)$ and computing $(2^{t+1}-1)$-expressions for a graph G of clique-width at most t [19]. We refer to the recent survey [20] for further information on different width parameters beyond tree-width.

There was an intensive study on the algorithmic perspective of graphs of bounded clique-width. There is a meta-theorem of Courcelle, Makowsky, and Rotics [5] that all problems expressible in MS_1-logic are fixed parameter tractable when parameterized by the clique-width of a graph. For many other problems, that are not expressible in this logic, like MAX-CUT, EDGE DOMINATING SET, GRAPH COLORING, or HAMILTONIAN CYCLE, there is a significant amount of the literature devoted to algorithms for these problems and their generalizations on graphs of bounded clique-width [9, 16, 15, 17, 22, 23, 24, 28, 29, 30]. The running time of all these algorithms on an n-vertex graph of clique-width at most t is $O(n^{f(t)})$, where f is some function of t.

One of the central questions in the area is whether the bound of $O(n^{f(t)})$ on the running time of all these algorithms is asymptotically optimal. Even the existence of fixed parameter tractable algorithms (with clique-width being the parameter) for all these problems (or their generalizations) was open until very recently [15, 22, 23, 24, 17]. As the first step toward obtaining lower bounds for clique-width parameterizations, we have shown in [13] that unless FPT$\neq$ W[1], there is no algorithm with run time $O(g(t) \cdot n^c)$, for some function g and a constant c not depending on t, for GRAPH COLORING, EDGE DOMINATING SET and HAMILTONIAN PATH.

Even though our results in [13] resolve the parameterized complexity of these problems, the conclusion that unless FPT$\neq$ W[1], there is no algorithm with run time $O(g(t) \cdot n^c)$, for some function g and a constant c not depending on t, is weak to compare the known algorithmic upper bounds. In this paper, we provide asymptotically tight optimal lower bounds for MAX-CUT and EDGE DOMINATING SET. In particular, we show that unless ETH fails, there is no $f(t)n^{o(t)}$-time algorithm for these problems, where f is an arbitrary function of k, on input of size n and clique-width at most k. While known algorithms for these problems run in times $n^{O(t^2)}$ [22, 23, 9, 30], we give new algorithmic upper bounds of the form $n^{O(t)}$. These two results together, lower and upper bounds, give asymptotically tight algorithmic bounds for MAX-CUT and EDGE DOMINATING SET.

To obtain our lower bounds we construct "linear FPT-reductions". These type of reductions are much more stringent and delicate than the usual FPT reductions. This is the reason why this research direction is still in a nascent stage and not so many asymptotically tight results are known in the literature. Chen et al. [2, 3] initiated this area of strong computational lower bounds and showed that there is no algorithm for k-CLIQUE (finding a clique of size k) running in time $f(k)n^{o(k)}$ unless there exists an algorithm for solving 3-SAT running in time $2^{o(n)}$ on a formula with n-variables. The assumption that there does not exists an algorithm for solving 3-SAT running in time $2^{o(n)}$ is known as EXPONENTIAL TIME HYPOTHESIS (ETH) [21] and it is equivalent to the parameterized complexity conjecture that FPT$\neq$M[1] [7, 12]. The lower bound on k-CLIQUE can be extended to some other parameterized

problems via a linear FPT-reductions [2, 3]. This kind of investigation has also been useful in obtaining tight algorithmic lower bounds for polynomial time approximation schemes [26] and for constraint satisfaction problems when parameterized by the tree-width of the "primal graph" [25]. We further extend the utility of this approach by obtaining asymptotically tight algorithmic bounds for clique-width parameterizations.

2 Definitions and Preliminary results

Parameterized Complexity. Parameterized complexity is a two dimensional framework for studying the computational complexity of a problem. One dimension is the input size n and another one is a *parameter* k. We refer to the books of Downey and Fellows [8] and Flum and Grohe [12] for a detailed treatment to parameterized complexity. Now we define the notion of parameterized (linear) reduction which is the main tool for establishing of our results.

Definition 1. *Let A, B be parameterized problems. We say that A is (uniformly many:1)* FPT***-reducible** *to B if there exist functions $f, g : \mathbb{N} \to \mathbb{N}$, a constant $\alpha \in \mathbb{N}$ and an algorithm Φ which transforms an instance (x, k) of A into an instance $(x', g(k))$ of B in time $f(k)|x|^\alpha$ so that $(x, k) \in A$ if and only if $(x', g(k)) \in B$. The reduction is called* linear *if $g(k) = O(k)$.*

Graphs: We only consider finite undirected graphs without loops or multiple edges. The vertex set of a graph G is denoted by $V(G)$ and its edge set by $E(G)$. A set $S \subseteq V(G)$ of pairwise adjacent vertices is called a *clique*. For $v \in V(G)$, by $E_G(v)$ we mean the set of edges incident to v. For a vertex v, we denote by $N_G(v)$ its *(open) neighborhood*, that is, the set of vertices which are adjacent to v. The *closed neighborhood* of v, that is, the set $N_G(v) \cup \{v\}$, is denoted by $N_G[v]$. The *degree* of a vertex v is denoted by $d_G(v)$. For a graph G, the incidence graph of G is the bipartite graph $I(G)$ with the vertex set $V(G) \cup E(G)$ such that $v \in V(G)$ and $e \in E(G)$ are adjacent if and only if v is incident to e in G.

Tree-width: A *tree decomposition* of a graph G is a pair (X, T) where T is a tree whose vertices we will call *nodes* and $X = (\{X_i \mid i \in V(T)\})$ is a collection of subsets of $V(G)$ such that

1. $\bigcup_{i \in V(T)} X_i = V(G)$,
2. for each edge $vw \in E(G)$, there is an $i \in V(T)$ such that $v, w \in X_i$, and
3. for each $v \in V(G)$ the set of nodes $\{i \mid v \in X_i\}$ forms a subtree of T.

The *width* of a tree decomposition $(\{X_i \mid i \in V(T)\}, T)$ equals $\max_{i \in V(T)}\{|X_i| - 1\}$. The *tree-width* of a graph G is the minimum width over all tree decompositions of G. We use notation $\mathbf{tw}(G)$ to denote the tree-width of a graph G.

Clique-width: Let G be a graph, and t be a positive integer. A t-graph is a graph whose vertices are labeled by integers from $\{1, 2, \ldots, t\}$. We call the t-graph consisting of exactly one vertex labeled by some integer from $\{1, 2, \ldots, t\}$ an initial t-graph. The *clique-width* $\mathbf{cwd}(G)$ is the smallest integer t such that G can be constructed by means of repeated application of the following four operations: (1) *introduce*: construction of an initial t-graph labeled by i (denoted by $i(v)$), (2) *disjoint union* (denoted by $\oplus$), (3) *relabel*: changing all labels i to j (denoted by $\rho_{i \to j}$) and (4) *join*: connecting all vertices labeled by i with all vertices labeled by j by edges (denoted by $\eta_{i,j}$).

An *expression tree* of a graph G is a rooted tree T of the following form:

- The nodes of T are of four types i, $\oplus$, η and ρ.
- Introduce nodes $i(v)$ are leaves of T, corresponding to initial t-graphs with vertices v, which are labeled i.
- A union node $\oplus$ stands for a disjoint union of graphs associated with its children.
- A relabel node $\rho_{i \to j}$ has one child and is associated with the t-graph, which is the result of relabeling operation for the graph corresponding to the child.

- A join node $\eta_{i,j}$ has one child and is associated with the t-graph, which is the result of join operation for the graph corresponding to the child.
- The graph G is isomorphic to the graph associated with the root of T (with all labels removed).

The *width* of the tree T is the number of different labels appearing in T. If a graph G has $\mathbf{cwd}(G) \leq t$ then it is possible to construct a rooted expression tree T with *width* t of G. Given a node X of an expression tree, the graph G_X represents the graph formed by the subtree of the expression tree rooted at X.

A well-known fact is that if the tree-width of a graph is bounded then its clique-width also is bounded. On the other hand, complete graphs have clique-width 2 and unbounded tree-width. But for sparse graphs the tree-width and clique-width are linearly related. Particularly, Gurski and Wanke [18] proved that if a graph G has no subgraph isomorphic to $K_{r,r}$, then $\mathbf{tw}(G) \leq 3(r-1)\mathbf{cwd}(G)-1$. Linear upper bounds of the clique-width by the tree-width for sparse graphs were established by Fomin et al. [14]. We use the following proposition.

Proposition 1 ([14]). *If G is a planar graph, then* $\mathbf{cwd}(G) \leq 12(\mathbf{tw}(G)+1)$.

Moreover, the proof is constructive and an expression tree for G of width at most $12(\mathbf{tw}(G)+1)$ can be constructed in FPT time (with tree-width being the parameter) from the tree decomposition of a planar graph G.

Capacitated Domination – Preliminary Results

A *capacitated graph* is a pair (G, c), where G is a graph and $c\colon V(G) \to \mathbb{N}$ is a *capacity* function such that $1 \leq c(v) \leq d_G(v)$ for every vertex $v \in V(G)$ (sometimes we simply say that G is a capacitated graph if the capacity function is clear from the context). A set $S \subseteq V(G)$ is called a *capacitated dominating set* if there is a *domination mapping* $f\colon V(G)\setminus S \to S$ which maps every vertex in $V(G)\setminus S$ to one of its neighbors such that the total number of vertices mapped by f to any vertex $v \in S$ does not exceed its capacity $c(v)$. We say that for a vertex $u \in S$, vertices in the set $f^{-1}(u)$ are *dominated by* u. In the CAPACITATED DOMINATING SET (or CDS) problem, we are given a capacitated graph (G, c) and a positive integer k as an input and the question is whether there exists a capacitated dominating set S for G containing at most k vertices.

We also consider a special variant of CDS problem which we call EXACT SATURATED CAPACITATED DOMINATING SET (or EXACT SATURATED CDS). Given a capacitated dominating set S, a vertex $v \in S$ is called *saturated* if the corresponding domination mapping f maps $c(v)$ vertices to v, that is, $|f^{-1}(v)| = c(v)$. A capacitated dominating set $S \subseteq V(G)$ is called *saturated* if there is a domination mapping f which saturates all vertices of S. In the EXACT SATURATED CAPACITATED DOMINATING SET problem, a capacitated graph (G, c) and a positive integer k are given as an input and the question is whether G has a saturated capacitated dominating set S with exactly k vertices.

A *red-blue capacitated graph* is a pair (G, c), where G is a bipartite graph with the vertex bipartition R and B and $c\colon R \to \mathbb{N}$ is a *capacity* function such that $1 \leq c(v) \leq d_G(v)$ for every vertex $v \in R$. The vertices of the set R are called *red* and the vertices of B are called *blue*. A set $S \subseteq R$ is called a *capacitated dominating set* if there is a *domination mapping* $f\colon B \to S$ which maps every vertex in B to one of its neighbors such that the total number of vertices mapped by f to any vertex $v \in S$ does not exceed its capacity $c(v)$. The RED-BLUE CAPACITATED DOMINATING SET (or RED-BLUE CDS) problem for a given red-blue capacitated graph (G, c) and a positive integer k, asks whether there exists a capacitated dominating set S for G containing at most k vertices. A capacitated dominating set $S \subseteq R$ is called *saturated* if there is a domination mapping f which saturates all vertices of S, that is, $|f^{-1}(v)| = c(v)$ for each $v \in S$. The RED-BLUE EXACT SATURATED DOMINATING SET problem (RED-BLUE EXACT SATURATED CDS) takes a red-blue capacitated graph (G, c) and a positive integer k as an input and asks whether there exists a saturated capacitated dominating set with exactly k vertices.

If the input graph G is restricted to be *planar* we call these problems PLANAR CDS, EXACT SATURATED PLANAR CDS, RED-BLUE PLANAR CDS and RED-BLUE EXACT SATURATED PLANAR CDS respectively. The following proposition can be deduced from the constructions presented in [1]. For completeness and ease of reference we provide a proof in the Appendix 6.1.

Proposition 2. PLANAR CDS, EXACT SATURATED PLANAR CDS, RED-BLUE PLANAR CDS *and* RED-BLUE EXACT SATURATED PLANAR CDS *can not be solved in time $f(t)n^{o(t)}$, where t is the tree-width of the input graph, unless ETH fails.*

The basic schema of all the proofs to come is following. The reduction in Proposition 2 is from k-CLIQUE to the above mentioned problems and the graphs obtained after the reduction are essentially $k \times k^2$ grid. This immediately implies that the tree-width t of these instances is $O(k)$. Now using the result of Chen et al. [2, 3] about k-CLIQUE, we conclude that PLANAR CDS, EXACT SATURATED PLANAR CDS, RED-BLUE PLANAR CDS and RED-BLUE EXACT SATURATED PLANAR CDS can not be solved in time $f(t)n^{o(t)}$, where t is the tree-width of the input graph, unless ETH fails. Now from Proposition 1 we know that the tree-width and the clique-width are linearly related in planar graph. This allows us to conclude that PLANAR CDS, EXACT SATURATED PLANAR CDS, RED-BLUE PLANAR CDS and RED-BLUE EXACT SATURATED PLANAR CDS can not be solved in time $f(t)n^{o(t)}$, where t is the *clique-width* of the input graph, unless ETH fails. This result is the starting point for our reductions to obtain the desired algorithmic lower bounds for MAX-CUT and EDGE DOMINATING SET when parameterized by the clique-width. Furthermore, our reductions are constrained to not blow up the clique-width in the resulting instances. That is, the clique-width of the input instance and the clique-width of the instance obtained after the reduction must be linearly related.

3 Max-Cut and related problems

In this section we consider the MAX-CUT problem and a few other problems that are closely related to it. A *cut set* of a graph G is the set of edges $C \subseteq E(G)$ such that the graph G' with the vertex set $V(G)$ and the edge set C is a bipartite graph. The size of a maximum cut set in G is denoted by $\mathbf{mcut}(G)$. For a partition V_1, V_2 of $V(G)$, the cut set is defined as $C_G(V_1, V_2) = \{uv \in E(G) \colon u \in V_1, v \in V_2\}$. It is well known that there is one to one correspondence between cut sets and partitions of the vertex set. In the MAX-CUT problem, we are given a graph G and a positive integer k, and the objective is to check whether there exists a cut set $C \subseteq E(G)$ such that $|C| \geq k$. Our main theorem in this section is following.

Theorem 1. *The* MAX-CUT *problem can not be solved in time $f(t)n^{o(t)}$ unless ETH fails. Moreover, the* MAX-CUT *problem can be solved in time $n^{O(t)}$. Here, t is the clique-width of the input graph.*

We prove this theorem in two parts. We first show the lower bound and then complement this result with the corresponding upper bound.

Lower Bounds. To prove our result we give a reduction from the RED-BLUE PLANAR CDS problem to the MAX-CUT problem. The proof is organized as follows: we first give a construction, then prove its correctness and finally argue on the clique-width of the transformed instance.
Construction: Let (G, c) be an instance of RED-BLUE PLANAR CDS with $R = \{u_1, \ldots, u_n\}$ being the set of red vertices and $B = \{v_1, \ldots, v_r\}$ being the set of blue vertices. We also assume that G has m edges and k is a positive integer. Now we describe the auxiliary gadgets.

Auxiliary gadgets* $F(x, y)$ *and* $F'(x, y)$*: Let x, y be two vertices. We construct $F(x, y)$ by joining x and y by $4m + 1$ paths of length two. The graph $F'(x, y)$ is constructed by joining x and y by $4m + 1$ paths of length three. The properties of $F(x, y)$ and $F'(x, y)$ which is required for our proof is summarized in the following simple lemma.

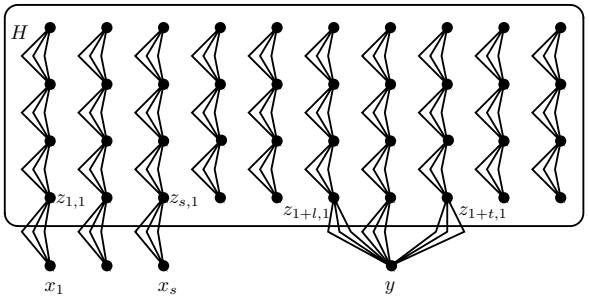

Figure 1: Graph $H_{s,t}(x_1, \ldots, x_s, y)$

Lemma 1. *For a pair of vertices x and y, $\mathbf{mcut}(F(x,y)) = 8m+2$, $\mathbf{mcut}(F'(x,y)) = 12m+3$. For any partition V_1, V_2 of the set of vertices in the gadget $F(x,y)$ such that $x \in V_1$ and $y \in V_2$, $|C_{F(x,y)}(V_1, V_2)| \leq \mathbf{mcut}(F(x,y)) - 4m - 1$, and for any partition V_1, V_2 of the set of vertices in the gadget $F'(x,y)$ such that $x, y \in V_1$, $|C_{F'(x,y)}(V_1, V_2)| \leq \mathbf{mcut}(F'(x,y)) - 4m - 1$.*

We are going to attach gadgets $F(x,y)$ and $F'(x,y)$ to other part of our construction through the vertices x and y. Notice that we can always assume that the vertices of $V(F(x,y)) \setminus \{x,y\}$ are included in exactly one side of an optimal partition of the vertex set leading to the maximum sized cut. Similarly, we can assume that the vertices of $N_{F'(x,y)}(x)$ ($N_{F'(x,y)}(y)$ respectively) also included in exactly one side of an optimal partition of the vertex set.

Auxiliary gadgets $H_{s,t}(x_1, \ldots, x_s, y)$**:** Let $l = \max\{n, r\}$. We first construct a graph H with the vertex set $\{z_{i,j} \colon 1 \leq i \leq 2l, 1 \leq j \leq 4m+1\}$. Any vertices $z_{i,j}$ and $z_{i',j'}$ are joined by an edge for $1 \leq i < i' \leq 2l$. That is, we get a complete $2l$ partite graph with the $2l$-partition $Z_1, \ldots, Z_{2l}$, where $Z_i = \{z_{i,1}, \ldots, z_{i,4m+1}\}$. Then we add graphs $F(z_{i,1}, z_{i,2}), \ldots, F(z_{i,4m}, z_{i,4m+1})$ for each $i \in \{1, \ldots, 2l\}$. Let $h = l(4m+1)(4ml + 16m + l)$. One can easily see that a partition V_1, V_2 corresponding to $\mathbf{mcut}(H)$ is following. Let V_1 consist of $Z_1, \ldots, Z_l$ and all the vertices of gadgets $F(z_{i,1}, z_{i,2}), \ldots, F(z_{i,4m}, z_{i,4m+1})$, $i \in \{l+1, \ldots, 2l\}$, except those vertices of these gadgets which are contained in $Z_{l+1}, \ldots, Z_{2l}$ and let V_2 be the remaining vertices. Using this partition V_1, V_2 corresponding to $\mathbf{mcut}(H)$ and Lemma 1, we get the following.

Lemma 2. *For any partition V_1, V_2 of the set of vertices of H such that if V_1 (or V_2) does not contain exactly l sets from $Z_1, \ldots, Z_{2l}$, then $|C_H(V_1, V_2)| \leq \mathbf{mcut}(H) - 4m - 1$. Furthermore, $\mathbf{mcut}(H) = h$.*

Let s and t be two positive integers such that $s, t \leq l$. We construct the graph $H_{s,t}(x_1, \ldots, x_s, y)$ from H by adding vertices $x_1, \ldots, x_s$ and y, and then joining them with H by the following gadgets, $F(x_1, z_{1,1}), \ldots, F(x_s, z_{s,1})$ and $F(y, z_{l+1,1}), \ldots, F(y, z_{l+t,1})$ (see Fig 1). Let $h_{s,t} = h + (8m+2)(s+t)$. Lemmata 1 and 2 imply the following properties of this graph.

Lemma 3. *The following properties holds for the graph $H_{s,t}(x_1, \ldots, x_s, y)$.*

- *The $\mathbf{mcut}(H_{s,t}(x_1, \ldots, x_s, y)) = h_{s,t}$.*
- *Let V_1, V_2 be an optimal partition of $V(H_{s,t}(x_1, \ldots, x_s, y))$, that is, $\mathbf{mcut}(H_{s,t}(x_1, \ldots, x_s, y)) = |C_{H_{s,t}(x_1, \ldots, x_s, y)}(V_1, V_2)|$, and $y \in V_1$. Then at most $l - t$ vertices among $x_1, \ldots, x_s$ are included in V_1.*
- *Furthermore, there is an optimal partition V_1, V_2 such that $y \in V_1$ and for any $0 \leq p \leq l - t$, exactly p vertices among $x_1, \ldots, x_s$ are included in V_1.*
- *Moreover, for any non optimal partition V_1, V_2 of $V(H_{s,t}(x_1, \ldots, x_s, y))$ such that (a) for any gadget $F(z_{i,j}, z_{i,j+1})$, $1 \leq i \leq 2l$ and $1 \leq j \leq 4m$, we have that $V(F(z_{i,j}, z_{i,j+1})) \setminus \{z_{i,j}, z_{i,j+1}\}$*

is either contained in V_1 or V_2; (b) for any gadget $F(x_i, z_{i,1})$, $1 \leq i \leq s$ and $F(y, z_{l+j,1})$ $1 \leq j \leq t$, we have that $V(F(x_i, z_{i,1})) \setminus \{x_i, z_{i,1}\}$ and $V(F(y, z_{l+j,1})) \setminus \{y, z_{l+j,1}\}$ is either contained in V_1 and V_2; then $|C_{H_{s,t}(x_1,\ldots,x_s,y)}(V_1, V_2)| \leq \mathbf{mcut}(H_{s,t}(x_1, \ldots, x_s, y)) - 4m - 1$.

Final Reduction: Now we describe our reduction. Each edge $e = u_i v_j$ of G is replaced by two vertices a_e and b_e and joined by edges to u_i and v_j. We create two vertices w_1 and w_2 and construct a copy of $F'(w_1, w_2)$. For each vertex $v_j \in B$, a copy of $F(v_j, w_1)$ is created. In the next step, we introduce a copy of $H_{n,l-k}(u_1, \ldots, u_n, w_1)$. By G' we denote the graph obtained until now. Finally, for each vertex $u_i \in R$, a copy of $H_{d_G(u_i),l-c(u_i)}(a_{e_1}, \ldots, a_{e_{d_G(u_i)}}, w_2)$, where $\{e_1, \ldots, e_{d_G(u_i)}\} = E_G(u_i)$ is constructed, and for each vertex $v_j \in B$, a copy of $H_{d_G(v_j),l-1}(a_{e_1}, \ldots, a_{e_{d_G(v_j)}}, w_2)$, where $\{e_1, \ldots, e_{d_G(v_j)}\} = E_G(v_j)$ is added. Let the resulting graph be Q. Let $\mu = (4m+1)(2r+3) + h_{n,l-k} + \sum_{i=1}^{n} h_{d_G(u_i),l-c(u_i)} + \sum_{j=1}^{r} h_{d_G(v_j),l-1} + 2(m+r)$.

Lemma 4. *The graph G has a capacitated dominating set of the size at most k if and only if Q has a cut set with at least μ edges.*

Proof. Let S be a capacitated dominating set of the size at most k in G and f be a corresponding domination mapping. We construct a partition V_1, V_2 of the vertex set of Q which corresponds to the cut set of size at least μ as follows. The vertex w_1 is included in V_1, the vertex w_2 is included in V_2, all vertices $v_1, \ldots, v_r$ are included in V_1, all the vertices in S are included in V_1 and vertices in $R \setminus S$ are included in V_2. We also include all the vertices b_e in V_2. For each edge $e = u_i v_j \in E(G)$ such that $f(v_j) = u_i$, that is, e is being used for domination, the corresponding vertex a_e is included in V_2 and all other vertices a_e, whose corresponding edge is not used for domination are included in V_1. Finally, we extend our partition to an optimal partition of all gadgets $F(x, y)$, $F'(x, y)$ and $H_{s,t}(x_1, \ldots, x_s, y)$ used in the construction of Q. The desired extensions of these gadgets to an optimal partition can be done by applications of Lemmata 1, 2 and 3. By construction of our partitions V_1 and V_2, the contribution of the gadgets $F(x, y)$, $F'(x, y)$ and $H_{s,t}(x_1, \ldots, x_s, y)$ to the cut $C_Q(V_1, V_2)$ is $\mathbf{mcut}(F(x, y))$, $\mathbf{mcut}(F'(x, y))$ and $\mathbf{mcut}(H_{s,t}(x_1, \ldots, x_s, y))$ respectively. Hence, we have already accounted for $(4m+1)(2r+3) + h_{n,l-k} + \sum_{i=1}^{n} h_{d_G(u_i),l-c(u_i)} + \sum_{j=1}^{r} h_{d_G(v_j),l-1}$ edges in the cut $C_Q(V_1, V_2)$. The remaining $2(m+r)$ edges in the cut $C_Q(V_1, V_2)$ come from the edges incident on the vertices a_e and b_e for some e. Look at an edge e, then we have two cases, either it is an edge used for domination or not. In the first case when $e = uv$ is used for dominating then ua_e, $a_e v$, ub_e and $b_e v$ are part of the cut. In the second case, for an edge when $e = uv$ exactly two of the edges among ua_e, $a_e v$, ub_e and $b_e v$ are part of the cut. In any case for every e at least two edges among ua_e, $a_e v$, ub_e and $b_e v$ are part of the cut and hence edges incident to the vertices a_e and b_e contribute at least $2(m-r) + 4r = 2(m+r)$ to the cut $C_Q(V_1, V_2)$. This completes the forward direction of the proof.

Assume now that Q has a cut set C of size at least μ, and V_1, V_2 be the corresponding partition of the vertex set of Q. Let Q' be the graph obtained by the union of the edge sets of auxiliary gadgets $F(x, y)$, $F'(x, y)$ and $H_{s,t}(x_1, \ldots, x_s, y)$. Then there exists a partition A and B of $V(Q')$ such that $C_{Q'}(A, B) = \mu'$, where $\mu' = (4m+1)(2r+3) + h_{n,l-k} + \sum_{i=1}^{n} h_{d_G(u_i),l-c(u_i)} + \sum_{j=1}^{r} h_{d_G(v_j),l-1}$. Suppose that at least for one of our auxiliary gadgets $F(x, y)$, $F'(x, y)$ or $H_{s,t}(x_1, \ldots, x_s, y)$, say $F(x, y)$, the partition V_1' and V_2' of $V(F(x, y))$ obtained by restricting the partition V_1 and V_2 to $V(F(x, y))$ is not optimal. That is, $|C_{F(x,y)}(V_1', V_2')| < \mathbf{mcut}(F(x, y))$. Then because of Lemmata 1, 2 and 3, $|C| \leq \mu' - (4m+1) + 4m < \mu$. By choosing a non-optimal partition of auxiliary gadgets we at least loose $4m+1$ edges while we can only gain $4m$ new edges by cutting $4m$ edges of Q which do not belong to these gadgets. This implies that C restricted to all these gadgets is an optimal cut in Q'. By Lemma 1, w_1 and w_2 belong to different sets of the bipartition V_1, V_2. Assume that $w_1 \in V_1$ and $w_2 \in V_2$. Then Lemma 1 implies that $v_1, \ldots, v_r \in V_1$. Thus, using Lemma 3 we conclude that at most k vertices of the set $R = \{u_1, \ldots, u_n\}$ belong to V_1. We set $S = R \cap V_1$ and prove that S is a capacitated dominating set in G. Notice that by Lemma 3, at most one vertex a_e in the neighborhood of each vertex v_j is included

in V_2. Suppose that there is a vertex v_j such that it's neighborhood in Q has no vertices $a_e \in V_2$. Then $|C| \leq \mu' + 2m + 2(r-1) < \mu$, a contradiction. So, for each vertex v_j, there is an edge $e = u_i v_j$ such that $a_e \in V_2$. Now we argue that $u_i \in S$. This follows from the fact that if $u_i \notin S$ then $u_i \in V_2$ and hence $|C| \leq \mu' + 2m + 2r - 2 < \mu$. We define the domination mapping $f(v_j) = u_i$. Since by Lemma 3, at most $c(u_i)$ vertices in the set $N_Q(u_i) \cap \{a_e \mid e \in E(G)\}$ are included in V_2, $|f^{-1}(u_i)| \leq c(u_i)$. This concludes the proof. □

Now we upper bound the clique-width of Q by a linear function of the tree-width of G.

Lemma 5. *Let $t \geq 2$. If $\mathbf{tw}(G) \leq t$ then $\mathbf{cwd}(Q) \leq 96t + 106$.*

Proof. Since $t \geq 2$, $\mathbf{tw}(I(G)) = \mathbf{tw}(G)$, and by Proposition 1 we have that $\mathbf{cwd}(I(G)) \leq c = 12t + 12$. We construct an expression tree for Q in two stages and use $8c + 10$ labels. At the first stage we construct en expression tree for G' using $4c + 10$ labels, and at the second stage we describe how it can modified to get an expression tree for Q using $4c$ additional labels.

Construction of an expression tree for G': Suppose that the expression tree for $I(G)$ uses c labels $\{\alpha_1, \ldots, \alpha_c\}$. To construct the expression tree for G' we use the following additional labels.

- Labels $\beta_1, \ldots, \beta_c$ for the vertices $v_1, \ldots, v_r$.
- Labels $\gamma_1, \ldots, \gamma_c$ for the vertices $\{a_e \mid e \in E(G)\}$.
- Labels $\delta_1, \ldots, \delta_c$ for the vertices $\{b_e \mid e \in E(G)\}$.
- Labels ζ_1, ζ_2 for the vertices w_1, w_2.
- Label η for the vertices $z_{i,j}$ in $H_{n,l-k}(u_1, \ldots, u_n, w_1)$.
- Working labels $\lambda_1, \lambda_2, \lambda_3$ and $\xi_1, \xi_2, \xi_3, \xi_4$.

We construct the required expression tree for G' by going over the expression tree for $I(G)$ and making necessary changes in it. When a vertex $u_i \in R$ labeled by α_p is introduced, we perform following set of operations. We first introduce the vertex u_i labeled α_p and a vertex (which is essentially $z_{i,1}$) labeled by ξ_3. Then $4m + 1$ vertices labeled with ξ_2 are introduced and joined with vertices labeled α_p and ξ_3. Then the vertices labeled ξ_2 are relabeled λ_1. Now we repeat the following operations $4m$ times: (a) introduce a vertex labeled ξ_1 and $4m + 1$ vertices labeled ξ_2; (b) join vertices labeled ξ_2 with vertices labeled ξ_1 and ξ_3; (c) relabel vertices labeled ξ_2 by λ_1, the vertex labeled ξ_3 by η, and the vertex labeled ξ_1 by ξ_3; (d) finally, the vertex labeled ξ_3 is relabeled η. We omit the union operations from our descriptions here and henceforth in any similar descriptions and assume that if some vertex is introduced then union is always performed.

When a vertex $x \in V(I(G))$ which corresponds to an edge $e \in E(G)$ labeled α_p is introduced, we introduce the vertices a_e and b_e and label it with γ_p and δ_p, respectively. Now we move toward introduction of vertices from the set B. When a vertex $v_j \in B$ labeled α_p is introduced, we introduce the vertex v_j with label β_p. Then $4m + 1$ vertices labeled ξ_1 are introduced, joined with the vertex labeled β_p and relabeled λ_2. We are labeling these vertices with λ_2 to finally join them with the vertex w_1, when it gets introduced.

For each union operation in the expression tree for $I(G)$, we do as follows. If both graphs contain vertices labeled η, then (a) vertices labeled η in one of the graphs are relabeled ξ_1; (b) we perform the union operation; (c) the vertices labeled η and ξ_1 are joined; and (d) the vertices labeled ξ_1 are relabeled η. If only one graph contains vertices labeled η then we just do the union operation.

If in the expression tree of $I(G)$, we have join operation between two labels say α_p and α_q then we simulate this by applying join operations between following: (i) α_p and γ_q; (ii) α_p and δ_q; (iii) β_p and γ_q; (iv) β_p and δ_q; (v) α_q and γ_p; (vi) α_q and δ_p; (vii) β_q and γ_p; and (viii) β_q and δ_p.

Finally, the relabel operation in the expression tree of G, that is, relabel α_p to α_q is replaced by following relabeling process: (a) α_p to α_q; (b) β_p to β_q; (c) γ_p to γ_q; and (d) δ_p to δ_q.

After we have completed the scanning of the expression tree for $I(G)$, the vertices w_1 and w_2 labeled by ζ_1 and ζ_2 respectively, are introduced. Then we repeat the following operations $4m+1$ times: (a) introduce two vertices labeled ξ_1 and ξ_2; (b) join vertices labeled ζ_1 and ξ_1, ξ_1 and ξ_2, ξ_2 and ζ_2; (c) and relabel vertices labeled ζ_1 and ζ_2 by λ_1. After that the vertex w_1 labeled ζ_1 is joined with vertices labeled λ_2.

Now we show how to complete the construction of $H_{n,l-k}(u_1,\ldots,u_n,w_1)$. We start of by repeating the following $l-n+k$ times. A vertex labeled ξ_3 is introduced. Now we repeat the following operations $4m$ times: (a) introduce a vertex labeled ξ_1 and $4m+1$ vertices labeled ξ_2; (b) join vertices labeled ξ_2 and vertices labeled with ξ_1 and ξ_3; (c) relabel vertices labeled ξ_2 by λ_1, the vertex labeled ξ_3 by ξ_4, and the vertex labeled ξ_1 by ξ_3. Finally, the vertex labeled ξ_3 is relabeled ξ_4, the vertices labeled ξ_4 are joined with vertices labeled η and then relabeled by η.

Now, we do the following $l-k$ times. A vertex labeled ξ_3 and $4m+1$ vertices labeled ξ_1 are introduced. The vertices labeled ξ_1 are joined with vertices labeled ζ_1 and ξ_2, and relabeled λ_1. After this we repeat the following operations $4m$ times: (a) introduce a vertex labeled ξ_1 and $4m+1$ vertices labeled ξ_2; (b) join vertices labeled ξ_2 and vertices labeled ξ_1 and ξ_3; (c) relabel vertices labeled ξ_2 by λ_1, the vertex labeled ξ_3 by ξ_4, and the vertex labeled ξ_1 by ξ_3. Finally, the vertex labeled ξ_3 is relabeled ξ_4, the vertices labeled ξ_4 are joined with vertices labeled η and then relabeled η.

Construction of an expression tree for Q: We now show how to modify the expression tree for G' to add gadgets $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ where $\{e_1,\ldots,e_{d_G(u_i)}\}=E_G(u_i)$ for $u_i\in R$ using $2c$ additional labels. The gadgets $H_{d_G(v_j),l-1}(a_{e_1},\ldots,a_{e_{d_G(v_j)}},w_2)$ where $\{e_1,\ldots,e_{d_G(v_j)}\}=E_G(v_j)$ for vertices $v_j\in B$ can be added in the same way by using additional $2c$ labels.

To add gadgets $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ where $\{e_1,\ldots,e_{d_G(u_i)}\}=E_G(u_i)$ for $u_i\in R$, we use following additional labels $\alpha'_1,\ldots,\alpha'_c$ and $\beta'_1,\ldots,\beta'_c$. We scan the expression tree for G' and iteratively change it for each u_i, $i\in\{1,\ldots,n\}$ to add the corresponding gadgets. Let $E_G(u_i)=\{e_1,\ldots,e_{d_G(u_i)}\}$. Let A denote the set of vertices $\{a_{e_1},\ldots,a_{e_{d_G(u_i)}}\}$ and let $U=A\cup\{u_i\}$. Let X be a node of the expression tree for G' and G'_X be the subgraph of G' corresponding to this node. If $V(G'_X)\cap U\neq\emptyset$ but $G'[U]$ is not a subgraph of G'_X then we can observe the following:

- If $u_i\in V[G'_X]$ then u_i is labeled by a label which is different from labels of other vertices of G'_X.
- If $u_i\notin V[G'_X]$ then vertices of $U\cap V(G'_X)$ are labeled by labels which are different from labels of other vertices of G'_X.

We use these observations to construct the graph $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ in such a way that all vertices $z_{i,j}$ of this gadget constructed for the node X are labeled by same labels, if $u_i\in V[G'_X]$ and this vertex is labeled by λ_p then we label it by α'_p, and if $u_i\notin V[G'_X]$ and the labels $\gamma_{p_1},\ldots,\gamma_{p_h}$ are used for vertices $U\cap V(G'_X)$ then all vertices $z_{i,j}$ are labeled by one of the labels $\gamma'_{p_1},\ldots,\gamma'_{p_h}$. The construction of the gadget $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ is completed when after some union operation all vertices of U are included in the graph G'_X. We finally relabel vertices of G'_X by the label λ_1 which is not used for any join operation.

When a vertex $u_i\in R$ labeled by α_p is introduced, we perform following set of operations. First, we introduce the vertex u_i labeled by α_p. Then we repeat the following operations $l+c(u_i)-d_G(u_i)$ times. A vertex labeled ξ_3 is introduced and then the following operations are repeated $4m$ times: (a) introduce a vertex labeled ξ_1 and $4m+1$ vertices labeled ξ_2, (b) join vertices labeled ξ_2 and vertices labeled ξ_1 and ξ_3, (c) relabel vertices labeled ξ_2 by λ_1, the vertex labeled ξ_3 by ξ_4, and the vertex labeled ξ_1 by ξ_3. Finally, the vertex labeled ξ_3 is relabeled ξ_4, the vertices labeled ξ_4 are joined with vertices labeled α'_p (if they exist) and then relabeled α'_p. Next we perform the following $l-c(u_i)$ times: (a) a vertex labeled ξ_3 and $4m+1$ vertices labeled ξ_1 are introduced; (b) the vertices labeled ξ_1 are joined with vertices labeled ξ_2, and relabeled λ_3. Then we repeat the following operations $4m$ times: (i) introduce a vertex labeled ξ_1 and $4m+1$ vertices labeled ξ_2, (ii) join vertices labeled ξ_2 and vertices labeled ξ_1 and ξ_3, (iii) relabel

vertices labeled ξ_2 by λ_1, the vertex labeled ξ_3 by ξ_4, and the vertex labeled ξ_1 by ξ_3. Finally, the vertex labeled ξ_3 is relabeled ξ_4, the vertices labeled ξ_4 are joined with vertices labeled α'_p and then relabeled α'_p.

When a vertex a_e such that $u_i a_e \in E(Q)$ labeled by γ_q is introduced, we perform following set of operations. First, we introduce the vertex a_e labeled γ_q and a vertex labeled by ξ_3. Then $4m+1$ vertices labeled ξ_2 are introduced and joined with vertices labeled γ_p and ξ_3. Then the vertices labeled ξ_2 are relabeled λ_1. Now we repeat the following operations $4m$ times: (a) introduce a vertex labeled ξ_1 and $4m+1$ vertices labeled ξ_2, (b) join vertices labeled ξ_2 and vertices labeled ξ_1 and ξ_3, (c) relabel vertices labeled ξ_2 by λ_1, the vertex labeled ξ_3 by γ'_q, and the vertex labeled ξ_1 by ξ_3. Finally, the vertex labeled ξ_3 is relabeled γ'_q.

Having dealt with introduction nodes, next we consider union operations. Let X be a union node of the expression tree for G'. Denote by X and Y two children of this node and let G'_Y and G'_Z be subgraphs of G' which correspond to these nodes. If one of these graphs do not contain vertices of U then we just perform the same operation. Otherwise we have two cases.

- $u_i \in V(G'_Y) \cup V(G'_Z)$. Suppose that $u_i \in V(G'_Y)$ and is labeled α_p. Then vertices $z_{i,j}$ of $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ which are constructed for the node Y are labeled α'_p. The graph G'_Z includes vertices of A labeled by some labels $\gamma_{p_1},\ldots,\gamma_{p_h}$, and all vertices $z_{i,j}$ of $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ corresponding to this node are labeled by same label γ'_{p_j}. We do the union operation as before, then join the vertices labeled α'_p and γ'_{p_j}, relabel the vertices labeled γ'_{p_j} by α'_p. If the graph G'_X corresponding to X contains all vertices of U, then the vertices labeled α'_p are relabeled λ_1.
- $u_i \notin V(G'_Y) \cup V(G'_Z)$. Then G'_Y includes vertices of A that are labeled by $\gamma_{p_1},\ldots,\gamma_{p_h}$, and all vertices $z_{i,j}$ of $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ corresponding to this node and are labeled by same label γ'_{p_i}. Similarly, G'_Z includes vertices of A labeled by some labels $\gamma_{q_1},\ldots,\gamma_{q_f}$, and all the vertices $z_{i,j}$ of $H_{d_G(u_i),l-c(u_i)}(a_{e_1},\ldots,a_{e_{d_G(u_i)}},w_2)$ corresponding to this node and are labeled by same label γ'_{q_j}. We relabel vertices labeled γ'_{p_i} by ξ_2 in the first graph, then perform the union operation, join the vertices labeled ξ_2 and γ'_{q_j}, relabel the vertices labeled ξ_2 by γ'_{q_j}.

The join operations in the expression tree for G' are done in the new tree in exactly the same way. The relabel operation in the expression tree of G', that is, relabel α_p to α_q and relabel γ_p to γ_q, are replaced by relabel α_p to α_q, α'_p to α'_q, and γ_p to γ_q, γ'_p to γ'_q, respectively.

When we have completed the scan of the expression tree for G', the only thing which remains is to join vertices labeled λ_3 and the vertex labeled ζ_2 (the vertex w_2). □

To conclude the first part of the proof of the Theorem 1, we observe that the number of vertices of Q is polynomial in $n+r$, and therefore if we could solve the MAX-CUT in time $f(t)|V(Q)|^{o(t)}$ where $t = \mathbf{cwd}(Q)$ then the RED-BLUE PLANAR CDS could be solved in time $f(t)|V(G)|^{o(t')} = f(t)|V(G)|^{o(t)}$ where $t' = \mathbf{tw}(G) = O(\mathbf{cwd}(Q)) = O(t)$.

Algorithmic upper bounds for Max-Cut. Now we outline an algorithm for solving MAX-CUT in time $n^{O(t)}$ on graphs of clique-width at most t. The algorithm is based on dynamic programming over the expression tree of the input graph. We briefly describe what we store in the tables corresponding to the nodes in the expression tree and move the further details to the Appendix 6.2.

Let G be a graph with n vertices and m edges, and let T be an expression tree for G of width t. For a node X of T, denote by G_X the t-graph associated with this node, and let $U_1(X),\ldots,U_t(X)$ be the sets of vertices of G_X labeled $1,\ldots,t$ respectively. The table of data for the node X stores vectors $(s_1,\ldots,s_t,r)$ of integers such that $0 \leq s_i \leq |U_i(X)|$ for $1 \leq i \leq t$, and $0 \leq r \leq |E(G_X)|$, for which there is a partition V_1, V_2 of $V(G_X)$ such that $|V_1 \cap U_i(G_X)| = s_i$ and $|C_{G_X}(V_1,V_2)| \geq r$. Notice that this table contains at most $(n+1)^t \cdot m$ vectors. If X is the root node of T (i.e. $G = G_X$) then $\mathbf{mcut}(G)$

is equal to the maximum value of r for which the table for X contains an entry with this value. This concludes the proof of the Theorem 1.

Theorem 1 have several interesting corollaries for similar problems like BIPARTIZATION BY EDGE REMOVAL and MAXIMUM (MINIMUM) BISECTION problem and they can be found in Appendix 6.3.

4 Edge Dominating Set

In this section, we consider the EDGE DOMINATING SET problem. In the EDGE DOMINATING SET problem, we are given a graph G and a positive integer k, and the objective is to determine whether there is a set of edges $X \subseteq E(G)$ such that $|X| \leq k$ and every edge of G is either included in X, or it is adjacent to at least one edge of X (which *dominates* it). The set X is called an *edge dominating set* of G. We prove the following result for EDGE DOMINATING SET.

Theorem 2. *The* EDGE DOMINATING SET *problem can not be solved in time $f(t)n^{o(t)}$ unless ETH fails. Moreover, the* EDGE DOMINATING SET *problem can be solved in time $n^{O(t)}$. Here, t is the clique-width of the input graph.*

The proof of Theorem 2 can be found in Appendix 6.4.

5 Conclusion and Further Directions

In this paper, we obtained the first asymptotically tight bounds for problems parameterized by the clique-width of the input graph. In particular, we showed that MAX-CUT and EDGE DOMINATING SET cannot be solved in time $f(t)n^{o(t)}$, unless ETH collapses; while there do exist algorithms with running time $n^{O(t)}$ for both these problems, where t is the clique-width of the input graph. We believe that our results opens a new direction in the algorithmic study around clique-width. Our reduction to obtain a tight lower bound for MAX-CUT is also an FPT-reduction, thus resolving an open problem about the parameterized complexity of MAX-CUT.

We conclude with an open problem related to HAMILTONIAN CYCLE. In the HAMILTONIAN CYCLE problem, we are given a graph G and the objective is to check whether there exists a cycle passing through every vertex of G. Similar to MAX-CUT and EDGE DOMINATING SET we can obtain the following algorithmic lower bound for the HAMILTONIAN CYCLE problem when parameterized by the clique-width of the input graph.

Theorem 3. *The* HAMILTONIAN CYCLE *problem can not be solved in time $f(t)n^{o(t)}$, where t is the clique-width of the input graph, unless ETH fails.*

A proof of Theorem 3 is given in Appendix 6.5. However, all the algorithms we know for HAMILTONIAN CYCLE run in time $n^{O(t^2)}$, where t is the clique-width of the input graph. We leave it open to find either an improved lower bound or an improved upper bound for the HAMILTONIAN CYCLE problem.

References

[1] H. BODLAENDER, D. LOKSHTANOV, AND E.PENNINKX, *Planar capacitated dominating set is $W[1]$-hard*, in IWPEC'09, 2009.

[2] J. CHEN, X. HUANG, I. A. KANJ, AND G. XIA, *On the computational hardness based on linear FPT-reductions*, J. Comb. Optim., 11 (2006), pp. 231–247.

[3] ——, *Strong computational lower bounds via parameterized complexity*, J. Comput. Syst. Sci., 72 (2006), pp. 1346–1367.

[4] D. G. CORNEIL, M. HABIB, J.-M. LANLIGNEL, B. A. REED, AND U. ROTICS, *Polynomial time recognition of clique-width $\leq$ 3 graphs (extended abstract)*, in LATIN'00, vol. 1776 of LNCS, Springer, 2000, pp. 126–134.

[5] B. COURCELLE, J. A. MAKOWSKY, AND U. ROTICS, *Linear time solvable optimization problems on graphs of bounded clique-width*, Theory Comput. Syst., 33 (2000), pp. 125–150.

[6] B. COURCELLE AND S. OLARIU, *Upper bounds to the clique width of graphs*, Discrete Appl. Math., 101 (2000), pp. 77–114.

[7] R. G. DOWNEY, V. ESTIVILL-CASTRO, M. R. FELLOWS, E. PRIETO, AND F. A. ROSAMOND, *Cutting up is hard to do: the parameterized complexity of k-cut and related problems*, Electr. Notes Theor. Comput. Sci., 78 (2003).

[8] R. G. DOWNEY AND M. R. FELLOWS, *Parameterized complexity*, Monographs in Computer Science, Springer-Verlag, New York, 1999.

[9] W. ESPELAGE, F. GURSKI, AND E. WANKE, *How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time*, in WG'01, vol. 2204 of LNCS, Springer, 2001, pp. 117–128.

[10] M. R. FELLOWS, D. HERMELIN, F. A. ROSAMOND, AND S. VIALETTE, *On the parameterized complexity of multiple-interval graph problems*, Theor. Comput. Sci., 410 (2009), pp. 53–61.

[11] M. R. FELLOWS, F. A. ROSAMOND, U. ROTICS, AND S. SZEIDER, *Clique-width minimization is NP-hard (extended abstract)*, in STOC'06, ACM, 2006, pp. 354–362.

[12] J. FLUM AND M. GROHE, *Parameterized complexity theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin, 2006.

[13] F. V. FOMIN, P. A. GOLOVACH, D. LOKSHTANOV, AND S. SAURABH, *Clique-width: on the price of generality*, in SODA, 2009, pp. 825–834.

[14] F. V. FOMIN, S.-IL OUM, AND D. M. THILIKOS, *Rank-width and tree-width of H-minor-free graphs*, Manuscript, (2008).

[15] M. U. GERBER AND D. KOBLER, *Algorithms for vertex-partitioning problems on graphs with fixed clique-width*, Theoret. Comput. Sci., 299 (2003), pp. 719–734.

[16] O. GIMÉNEZ, P. HLINENÝ, AND M. NOY, *Computing the tutte polynomial on graphs of bounded clique-width*, SIAM J. Discret. Math., 20 (2006).

[17] B. GODLIN, T. KOTEK, AND J. A. MAKOWSKY, *Evaluations of graph polynomials*, in WG, vol. 5344 of Lecture Notes in Computer Science, 2008, pp. 183–194.

[18] F. GURSKI AND E. WANKE, *The tree-width of clique-width bounded graphs without $k_{n,n}$*, in WG'00, LNCS, Springer, 2000, pp. 196–205.

[19] P. HLINENÝ AND S. IL OUM, *Finding branch-decompositions and rank-decompositions*, SIAM J. Comput., 38 (2008), pp. 1012–1032.

[20] P. HLINENÝ, S.-IL OUM, D. SEESE, AND G. GOTTLOB, *Width parameters beyond tree-width and their applications*, Comput. J., 51 (2008), pp. 326–362.

[21] R. IMPAGLIAZZO, R. PATURI, AND F. ZANE, *Which problems have strongly exponential complexity?*, J. Comput. Syst. Sci., 63 (2001), pp. 512–530.

[22] D. KOBLER AND U. ROTICS, *Polynomial algorithms for partitioning problems on graphs with fixed clique-width (extended abstract)*, in SODA'01, ACM-SIAM, 2001, pp. 468–476.

[23] ——, *Edge dominating set and colorings on graphs with fixed clique-width*, Discrete Appl. Math., 126 (2003), pp. 197–221.

[24] J. MAKOWSKY, U. ROTICS, I. AVERBOUCH, AND B. GODLIN, *Computing graph polynomials on graphs of bounded clique-width*, in WG'06, LNCS, Springer, 2006, pp. 191–204.

[25] D. MARX, *Can you beat treewidth?*, in FOCS, 2007, pp. 169–179.

[26] ——, *On the optimality of planar and geometric approximation schemes*, in FOCS, 2007, pp. 338–348.

[27] S.-IL OUM AND P. SEYMOUR, *Approximating clique-width and branch-width*, J. Combin. Theory Ser. B, 96 (2006), pp. 514–528.

[28] M. RAO, *MSOL partitioning problems on graphs of bounded treewidth and clique-width*, Theoret. Comput. Sci., 377 (2007), pp. 260–267.

[29] K. SUCHAN AND I. TODINCA, *On powers of graphs of bounded NLC-width (clique-width)*, Discrete Appl. Math., 155 (2007), pp. 1885–1893.

[30] E. WANKE, *k-NLC graphs and polynomial algorithms*, Discrete Appl. Math., 54 (1994), pp. 251–266.

6 Appendix

6.1 Proof of Proposition 2

In the proof that PLANAR CDS is $W[1]$-hard when parameterized by the tree-width presented in [1], the reduction is a polynomial time linear FPT-reduction from the k-MULTI-COLORED CLIQUE (k-MCC) problem. Recall that the k-MCC asks for a given k-partite graph $G = (V_1 \cup \cdots \cup V_k, E)$, where $V_1, \ldots, V_k$ are sets of the k-partition, whether there is a k-clique C in G. The fact that this problem can not be solved in time $f(k)n^{o(k)}$ unless ETH fails follows immediately from the reduction from the k-CLIQUE problem (see e.g. [10]). The reduction to PLANAR CDS goes via an intermediate problem, called PLANAR ARC SUPPLY (PAS) which we describe here. In PAS we are given a planar digraph $D = (N, A)$ with $|N(D)| + |A(D)| = k$, no loops and no double arcs. Every node $u \in N$ has a demand $\zeta(u)$ and every arc $uv \in A$ has a list $L(uv)$ of ordered integer pairs, called the *supply pairs* of uv. The supply pairs and the demand are all coded in unary. The task is to decide whether there is a function $f_a : A \to \mathbb{N}$ and a function $f_b : A \to \mathbb{N}$ such that for every arc $uv \in A$ we have $(f_a(uv), f_b(uv)) \in L(uv)$ and for every node $u \in N$ we have that $\zeta(u) \leq \sum_{v \in N^+(u)} f_a(uv) + \sum_{v \in N^-(u)} f_b(vu)$. In essence we are asked to pick a supply pair from the list of every arc such that for every vertex the arcs incident to it are able to cover the demand of the vertex. Therefore the pair of functions f_a and f_b are called a *supply selection* and a supply selection is called *satisfying* if the demand of all vertices is met.

In [1] a reduction from k-MCC to PAS is given. Let D be the directed graph of the PAS instance constructed from a an instance of k-MCC and let the underlying undirected graph of D be $U(D)$. Then D has the following key properties, (a) $U(D)$ is a subgraph of a $O(k) \times O(k^2)$ grid (and hence has tree-width $O(k)$), (b) every vertex of D either has in-degree 1 and out-degree 1 or in-degree 2 and out-degree 2, and (c) the distance in $U(D)$ between any two vertices of degree 4 is at least 3. Lemmata 1 and 2 from [1] imply that PAS restricted to digraphs with properties (a),(b) and (c) can not be solved in time $f(t)n^{o(t)}$ unless ETH collapses, where t is the tree-width of the underlying undirected graph $U(D)$ of the input digraph D. Another property of the constructed PAS instances is that whenever they are yes-instances there is a satisfying supply selection such that for every vertex the supply exactly equals the demand. Therefore, the exact version of PAS, (EPAS) where the supply is required to be exactly the demand can not be solved in time $f(t)n^{o(t)}$ unless ETH fails as well.

We now give a reduction from EPAS (with properties (a),(b),(c)) to EXACT SATURATED PLANAR CDS. We consider here a slightly modified version of EXACT SATURATED PLANAR CDS, which we call EXACT SATURATED PLANAR MARKED CDS, where we *mark* some vertices and demand that all marked vertices must be in the dominating set. In particular we show how to transform an instance D, k of EPAS where $U(D)$ is a subgraph of a $t \times t^2$ grid, to an instance H, k^* of EXACT SATURATED PLANAR MARKED CDS. Let $\mathcal{X}$ be the largest demand of any vertex of D and $\mathcal{U} \leq \mathcal{M} \leq \mathcal{D}$ be large positive integers, chosen such that both $\mathcal{U}$ and $\mathcal{M} - \mathcal{U}$ are an order of magnitude (at least a 100) times larger than $\mathcal{X}$. Modify the demand of each vertex v of D by increasing it by $d(v)M$. Modify every supply-pair (a, b) to $(\mathcal{U} + a, \mathcal{D} + b)$. It is easy to see that by property (b) these modifications do not change whether D,k is a yes instance of PAS.

To build H we start with the node set $N(D)$ and make every vertex of $N(D)$ marked. For every arc uv of D we make a gadget between u and v in H. In particular, for an arc $uv \in A(D)$, for every pair of integers $(p, q) \in L(u, v)$ we add a vertex w to H, make w adjacent to u, add p vertices of degree 2 adjacent to u and w and add q vertices of degree 2 adjacent to w and v. We call the vertex w is a *list* vertex. This concludes the construction of the graph H. Since D is planar and the gadget we add to H for every arc of D is planar, H is planar as well. Every marked vertex v of H is also a vertex in D. The capacity of v in H is set to $d_H(v) - \zeta(v) - d_D^+(v)$, that is, the degree of v in H, minus v's demand in D and minus v's out-degree in D. For all unmarked vertices, their capacity in H is equal to their degree in H. Finally, $k^* = |N(D)| + |A(D)|$. This concludes the construction of the EXACT SATURATED PLANAR MARKED CDS instance (H, k^*). Observe that H can be obtained from $U(D)$

by subdividing edges, and adding copies of subdivided edges. Any sequence of these operations can increase the tree-width of a graph by at most 1 and hence the tree-width of H is at most $t + 1$.

Claim 1. *If D has a satisfying supply selection then H has a saturated capacitated dominating set of size k^**

Proof. We build a capacitated dominating set S of H. First we insert all the marked vertices of H in S. For every arc uv of D we add a list vertex w to S, namely the list vertex that corresponds to the supply pair in $L(uv)$ that was selected by the satisfying supply selection of D. The size of S is $|N(D)| + |A(D)| = k^*$. We now prove that S is a capacitated dominating set of H.

First, observe that the marked vertices of H form a dominating set of H, so S is a dominating set of H. Now, every unmarked vertex in S has capacity equal to its degree, so all unmarked vertices in S dominate all their neighbors. We now prove that for every marked vertex u, the number of yet undominated neighbors of u is at most the capacity of u. The number of neighbors of u that already have been dominated is at least $\zeta(u)$. The number of neighbors of u that are in S is $d_D^+(u)$. Hence, the total number of yet undominated neighbors of u is at most $d_H(u) - \zeta(u) - d_D^+(u)$ which is exactly the capacity of u. Hence S is a saturated capacitated dominating set of H of size k^*. □

Claim 2. *If H has a capacitated dominating set S of size k^* then D has a satisfying supply selection.*

Proof. There are two kinds of unmarked vertices in H, list vertices and vertices of degree 2. Every degree 2 vertex u has exactly one neighbor that is unmarked, and one neighbor v that is a list vertex. Since the capacity of v is equal to its degree and all marked vertices must be in S, if $u \in S$ then $S \cup \{v\} \setminus \{u\}$ is a capacitated dominating set of H of size at most k^*. Thus, without loss of generality, all unmarked vertices in S are list vertices.

For an arc uv of D, let $s(uv)$ be the number of vertices in S in the gadget corresponding to the arc uv. For a vertex u of D let $s^+(u) = \sum_{uv \in A(D)} s(uv)$, $s^-(u) = \sum_{vu \in A(D)} s(uv)$ and $s(u) = s^+(u) + s^-(u)$. Since S contains at most $|A(D)|$ unmarked vertices we have that $\sum_{u \in V(D)} s(u) \leq 2|A(D)|$. If $s(u) < d_D(u)$ for a vertex u then the number of vertices in $N_H(u)$ dominated by vertices other than u is at most $s(u) \cdot (\mathcal{D} + 4\mathcal{X}) < d_D(u)M$. However the capacity of u is at most $d_H(u) - d_D(u)M$, contradicting that S is a capacitated dominating set. Hence, for every node $u \in N(D)$, $s(u) \geq d_D(u)$. If for some node $s(u) > d_D(u)$ then $\sum_{u \in N(D)} s(u) > \sum_{u \in N(D)} d_D(u) = 2|A(D)|$, contradicting that $\sum_{u \in N(D)} s(u) \leq 2|A(D)|$. Thus, for every node $u \in N(D)$, $s(u) = d_D(u)$.

Consider now three consecutive arcs pq, qr and rs in $A(D)$ such that both q and r have degree 2 in D. There are three cases, either $s(pq) = s(qr) = s(qs) = 1$ or $s(pq) = s(qs) = 2$ and $s(qr) = 0$ or finally $s(pq) = s(qs) = 0$ and $s(qr) = 2$. We show that the last two cases lead to a contradiction. If $s(pq) = s(qs) = 2$ and $s(qr) = 0$ then the number of neighbors of r dominated by vertices other than r is at most $2(\mathcal{U} + 4\mathcal{X}) < 2M$. However the capacity of r is at most $d_H(r) - 2M$, contradicting that S is a capacitated dominating set. Similarly, if $s(pq) = s(qs) = 0$ and $s(qr) = 2$ then the number of neighbors of q dominated by vertices other than q is at most $2(\mathcal{U} + 4\mathcal{X}) < 2M$. However the capacity of q is at most $d_H(q) - 2M$, contradicting that S is a capacitated dominating set. It follows that $s(pq) = s(qr) = s(qs) = 1$. Because the distance in H between any pair of vertices with degree 4 is at least 3 it follows that $s(pq) = 1$ for every arc $pq \in A(D)$.

We now make a supply selection (f_a, f_b) for D as follows. For every arc uv there is exactly one unmarked vertex x in S in the gadget in H corresponding to the arc uv. This vertex x corresponds to a pair $(p, q) \in L(uv)$ and we make uv select the pair (p, q). Every arc selects a pair from its list in this manner. We now show that this supply selection is satisfying. Suppose for contradiction that this is not the case, then there is some vertex $u \in N(D)$ whose demand is not met. Then u is a marked vertex in H, and the demand of u is $d_H(u) - \zeta(u) - d_D^+(u)$. The number of neighbors of u that are dominated by vertices other than u is at most $\sum_{v \in N^+(u)} f_a(uv) + \sum_{v \in N^-(u)} f_b(vu) < \zeta(u)$. Since $s(pq) = 1$ for every arc $pq \in A(D)$, u is adjacent to exactly $d^+(u)$ vertices in S. Thus u must dominate more than $d_H(u) - \zeta(u) - d_D^+(u)$ vertices, a contradiction. This concludes the proof. □

The construction, together with Claims 2 and 1 imply that EXACT SATURATED PLANAR CDS and PLANAR CDS can not be solved in time $f(t)n^{o(t)}$ unless ETH collapses. If we in the construction color all marked vertices and all list vertices red, all other vertices blue and decrease the capacity of every marked vertex by the number of list vertices in its neighborhood minus 1, Claims 2 and 1 also go through for RED-BLUE EXACT SATURATED PLANAR CDS and RED-BLUE PLANAR CDS. Hence RED-BLUE PLANAR CDS and RED-BLUE EXACT SATURATED PLANAR CDS can not be solved in time $f(t)n^{o(t)}$ unless ETH fails. This concludes the proof.

6.2 Algorithmic upper bounds for Max-Cut.

Now we outline an algorithm for solving MAX-CUT in time $n^{O(t)}$ on graphs of clique-width at most t. The algorithm is based on dynamic programming over the expression tree of the input graph. We first describe what we store in the tables corresponding to the nodes in the expression tree. and move the further

Let G be a graph with n vertices and m edges, and let T be an expression tree for G of width t. For a node X of T, denote by G_X the t-graph associated with this node, and let $U_1(X), \ldots, U_t(X)$ be the sets of vertices of G_X labeled $1, \ldots, t$ respectively. The table of data for the node X stores vectors $(s_1, \ldots, s_t, r)$ of integers such that $0 \leq s_i \leq |U_i(X)|$ for $1 \leq i \leq t$, and $0 \leq r \leq |E(G_X)|$, for which there is a partition V_1, V_2 of $V(G_X)$ such that $|V_1 \cap U_i(G_X)| = s_i$ and $|C_{G_X}(V_1, V_2)| \geq r$. Notice that this table contains at most $(n+1)^t \cdot m$ vectors. If X is the root node of T (that is, $G = G_X$) then $\mathbf{mcut}(G)$ is equal to the maximum value of r for which the table for X contains an entry with this value.

Now we give the details of how we make our tables and how do we update it.

Introduce Node: Tables for introduce nodes of T are constructed in a straightforward manner.

Relabel Node: Suppose that X is a relabel node $\rho_{i\to j}$, and let Y be the child of X. Then the table for X contains a vector $(s_1, \ldots, s_t, r)$ if and only if $s_i = 0$ and the table for Y contains the entry $(s'_1, \ldots, s'_t, r)$ such that $s'_p = s_p$ for $1 \leq p \leq t$, $t \neq i, j$, and $s_j = s'_i + s'_j$.

Union Node: Let X be a union node with children Y and Z. In this case the table for X contains a vector $(s_1, \ldots, s_t, r)$ if and only if the tables for Y and Z have vectors $(s'_1, \ldots, s'_t, r')$ and $(s''_1, \ldots, s''_t, r'')$ respectively, such that $s'_i + s''_i = s_i$ for $1 \leq i \leq t$, and $r' + r'' \geq r$.

Join Node: Finally, suppose that X is a join node $\eta_{i,j}$ with the child Y. It can be noted that the table for X has a vector $(s_1, \ldots, s_t, r)$ if and only if the table for Y includes a vector $(s_1, \ldots, s_t, r')$ such that $r' + s_i(|U_j(Y)| - s_j) + s_j(|U_i(Y)| - s_i) \geq r$.

Correctness of the algorithm follows from the description of the procedure, and its runs in time $O(t^{O(1)}n^{2t+O(1)})$. This proves that MAX-CUT can be solved in time $n^{O(t)}$ on graphs of clique-width at most t.

6.3 Bipartization by Edge Removal and Maximum (Minimum) Bisection

In the BIPARTIZATION BY EDGE REMOVAL problem, we are given a graph G and a positive integer k, and the question is whether there is a set of edges X such that $|X| \leq k$ and the graph G' with the vertex set $V(G)$ and the edge set $E(G) \setminus X)$ is bipartite. Since this problem is dual to the MAXIMUM CUT problem, we immediately have the following corollary.

Corollary 1. *The* BIPARTIZATION BY EDGE REMOVAL *problem can not be solved in time $f(t)n^{o(t)}$ unless ETH fails. Moreover, the* BIPARTIZATION BY EDGE REMOVAL *problem can be solved in time $n^{O(t)}$. Here, t is the clique-width of the input graph.*

In the MAXIMUM (MINIMUM) BISECTION problem, we are given a graph G with an even number of vertices and a positive integer k, and the objective is to check whether there is a partition of $V(G)$ into two sets V_1 and V_2 of equal size such that $|C_G(V_1, V_2)| \geq k$ ($|C_G(V_1, V_2)| \leq k$).

Corollary 2. *The* MAXIMUM (MINIMUM) BISECTION *problem can not be solved in time $f(t)n^{o(t)}$ unless ETH fails. Moreover, the* MAXIMUM (MINIMUM) BISECTION *problem can be solved in time $n^{O(t)}$. Here, t is the clique-width of the input graph.*

Proof. The algorithmic upper bound for the MAXIMUM BISECTION follows from the observation that the algorithm for the MAX-CUT described in 6.2 can be modified for this problem. The lower bound can be obtained from the fact that the MAX-CUT problem for a graph G can be reduced to the MAXIMUM BISECTION by adding $|V(G|$ isolated vertices. The claim about the MINIMUM BISECTION follows from the observation that the MAXIMUM BISECTION problem for a graph G can be reduced to the MINIMUM BISECTION problem in the complement $\overline{G}$, and the fact that $\mathbf{cwd}(\overline{G}) \leq 2\mathbf{cwd}(G)$ (see [30, 6]). □

6.4 Proof of Theorem 2

We prove this theorem in two parts. We first show the lower bound and then complement this result with the corresponding upper bound. Our proof for lower bounds uses several ideas form the proof of W[1]-hardness given in [13], though requires several new ideas to get a linear bound on the clique-width of the instances we reduce to.

Lower Bounds. To prove our result we give a linear FPT-reduction from the RED-BLUE EXACT SATURATED PLANAR CDS problem to the EDGE DOMINATING SET problem. The proof is organized as follows: we first give a construction, then prove its correctness and finally argue on the clique-width of the transformed instance. We start with descriptions of auxiliary gadgets.

Auxiliary gadgets: Let $s \leq t$ be positive integers. We construct a graph $F_{s,t}$ with the vertex set $\{x_1, \ldots, x_s, y_1, \ldots, y_s, z_1, \ldots, z_t\}$ and edges x_iy_i, $1 \leq i \leq s$ and y_iz_j, $1 \leq i \leq s$ and $1 \leq j \leq t$. Basically we have complete bipartite graph between y_i's and z_j's with pendent vertices attached to y_i's. The vertices $z_1, z_2, \ldots, z_t$ are called *roots* of $F_{s,t}$.

Graph $F_{s,t}$ has the following property.

Lemma 6. *Any set of s edges incident to vertices $y_1, \ldots, y_s$ forms an edge dominating set in $F_{s,t}$. Furthermore, let G be a graph obtained by the union of $F_{s,t}$ with some other graph H such that $V(F_{s,t}) \cap V(H) = \{z_1, \ldots, z_t\}$. Then every edge dominating set of G contains at least s edges from $F_{s,t}$.*

The proof of the lemma follows from the fact that every edge dominating set includes at least one edge from $E(y_i)$ for $i \in \{1, \ldots, s\}$.

Final Reduction: Now we describe our reduction. Let (G, c) be red-blue capacitated graph with $R = \{u_1, \ldots, u_n\}$ being the set of red vertices and $B = \{v_1, \ldots, v_r\}$ being the set of blue vertices and k be a positive integer. Each red vertex u_i is replaced by the set U_i with $c(u_i)$ vertices, and for every edge $u_iu_j \in E(G)$, all vertices of U_i are joined to v_j by edges. For every vertex v_j we add one additional leaf (a pendent vertex). Now vertex sets $\{a_1, \ldots, a_n\}$ and $\{b_1, \ldots, b_n\}$ are constructed, and vertices a_i are made adjacent to all the vertices of U_i and the vertex b_i. For every vertex b_i, a set R_i of $c(u_i)$ vertices is added and b_i is made adjacent to all the vertices in R_i. Then to every vertex of $R_1 \cup R_2 \cup \cdots \cup R_n$ we add a path of length two. Let X be the set of middle vertices of these paths. We denote the obtained graph by G' (see Fig 2). Finally, we introduce three copies of $F_{s,t}$:

- a copy of $F_{n-k,n}$ with roots $\{a_1, \ldots, a_n\}$,
- a copy of $F_{k,n}$ with roots $\{b_1, \ldots, b_n\}$, and a

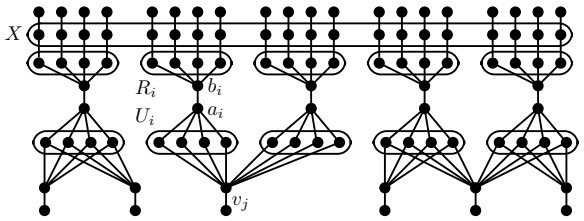

Figure 2: Graph G'

- a copy of $F_{m,r}$ where $r = \sum_{i=1}^{n} c(u_i)$ with roots in X.

Let this final resulting graph be H.

Lemma 7. *A graph G has a saturated capacitated dominating set of the size k if and only if H has an edge dominating set of cardinality at most $n + m + r$.*

Proof. Let S be an exact saturated dominating set of the size k in G and f be its corresponding domination mapping. For convenient, we assume that $S = \{u_1, \ldots, u_k\}$. We construct the edge dominating set as follows. First we select an edge emanating from every vertex in the set $\{v_1, \ldots, v_m\}$. For every vertex v_j, we choose a vertex u in U_i where $u_i = f(v_j)$ which is not incident to already chosen edges and add the edge uv_i to our set. Notice that we always have such a choice of $u \in U_j$ as $c(u_j) = |U_j|$. We observe that these edges already dominate all the edges in the sets $E(v_j)$, $1 \leq j \leq m$, and in sets $E(u)$ for $u \in U_1 \cup \cdots \cup U_k$. Now we add $n - k$ edges from $F_{n-k,n}$ which are incident to vertices in $\{a_{k+1}, \ldots, a_n\}$ and k edges from $F_{k,n}$ which are incident to $\{b_1, \ldots, b_k\}$. Then $r - m$ matching edges joining vertices of $R_{k+1}, \ldots, R_n$ to the vertices of X are included in the set. Finally, we add m edges form $F_{m,r}$ which are incident to vertices of X and are adjacent to vertices of $R_1, \ldots, R_k$. Since S is an exact capacitated dominating set, $\sum_{i=1}^{k} c(u_i) = m$, and from our description it is clear that the resulting set is an edge dominating set of size $n + m + r$ for H.

We proceed to prove the other direction of the equivalence. Let L be an edge dominating set of cardinality at most $n+m+r$. The set L is forced to contain at least one edge from every $E(v_j)$, at least $n - k$ edges from $F_{n-k,n}$, at least k edges from $F_{k,n}$, and at least one edge from $E(x)$ for all $x \in X$, because of the presence of pendent edges. This implies that $|L| = n+m+r$, and L contains exactly one edge from every $E(v_i)$, exactly $n-k$ edges from $F_{n-k,n}$, exactly k edges from $F_{k,n}$, and exactly one edge from $E(x)$ for all $x \in X$. Every edge a_ib_i needs to be dominated by some edge of L, in particular it must be dominated from either an edge of $F_{n-k,n}$, or $F_{k,n}$. Let $I = \{i : a_i \text{ is incident to an edge from } L \cap E(F_{n-k,n})\}$ and $J = \{j \colon b_j \text{ is incident to an edge from } L \cap E(F_{k,n})\}$. The above constraints on the set L implies that $|I| = n - k$, $|J| = k$, and these sets form a partition of $\{1, \ldots, n\}$. The edges which join vertices b_i and R_i for $i \in I$ are not dominated by edges from $L \cap E(F_{k,n})$. Hence to dominate these edges we need at least $\sum_{i \in I} |R_i|$ edges which connect sets R_i and X. Since at least m edges of $F_{m,r}$ are included in L, we have that $\sum_{i \in I} |R_i| \leq r - m$ and $\sum_{j \in J} |R_j| = r - \sum_{i \in I} |R_i| \geq r - (r - m) \geq m$. Let $S = \{u_i \colon i \in J\}$. Clearly, $|S| = k$. Now we show that S is a saturated capacitated dominating set. For $i \in J$, edges which join a vertex a_i to U_i are not dominated by edges from $L \cap E(F_{n-k,k})$, and hence they have to be dominated by edges from sets $E(v_j)$. Since $m \leq \sum_{i \in J} |R_j| = \sum_{i \in J} |U_j|$, there are exactly m such edges, and every such edge must be dominated by exactly one edge from L. We also know that $L \cap E(v_j) \neq \emptyset$ for all $j \in \{1, \ldots, m\}$ and hence for every v_j, there is exactly one edge which joins it with some vertex $u \in U_i$ for some $i \in J$. Furthermore, all these edges are not adjacent, that is, they form a matching. We define $f(v_j) = u_i$ for $j \in \{1, \ldots, m\}$. From our construction it follows that f is a domination mapping for S and S is an exact saturated dominating set in G. □

The next lemma shows that if the graph G we started with has bounded tree-width then H has bounded clique-width.

Lemma 8. *If* $\mathbf{tw}(G) \leq t$ *then* $\mathbf{cwd}(H) \leq 12t + 24$.

Proof. By Proposition 1, we have that the graph G is of clique-width at most $s = 12t + 12$. Suppose that the expression tree for G uses s-labels $\{\alpha_1, \ldots, \alpha_s\}$. We construct the expression tree for H by scanning the expression tree for G. To construct the expression tree for H we need following additional labels:

- Labels ξ_1, ξ_2, and ξ_3 for attaching $F_{n-k,n}$, $F_{k,n}$ and $F_{m,r}$ respectively.
- Labels $\zeta_1, \zeta_2, \zeta_3$ for marking some vertices.
- Working labels $\gamma_1, \ldots, \gamma_6$.

When a vertex $u_i \in R$ labeled α_p is introduced, we perform following set of operations. First we introduce following vertices with some working labels: $c(u_i)$ vertices of U_i with label γ_1, the vertex a_i with label γ_2, and the vertex b_i with label γ_3. Then we join the vertices labeled with γ_1 to the vertex labeled with γ_2, and the vertex labeled γ_2 with the vertex labeled γ_3. Now we want to make the vertices of R_i and the paths attached to it. To do so we perform following operations $c(u_i)$ times: introduce three nodes labeled with γ_4, γ_5 and γ_6 and join γ_3 with γ_4, γ_4 with γ_5 and γ_5 with γ_6. Finally, we relabel γ_1 to α_p, γ_2 to ξ_1 and γ_3 to ξ_3, γ_4 to ζ_1, γ_5 to ξ_3, and γ_6 to ζ_2. We omit the union operations from the description and assume that if some vertex is introduced then this operation is immediately performed.

If a vertex $v_j \in B$ labeled α_q is introduced then we introduce the vertex v_i labeled by γ_1, and the vertex labeled γ_2 (pendent vertex), join vertices labeled γ_1 and γ_2, and then relabel γ_1 to α_q and γ_2 to ζ_3.

If in the expression tree of G, we have join operation between two labels say α_p and α_q then we repeat in the new tree. Union operations in the expression tree is exactly done as before.

Finally to complete the expression tree for H, we need to add $F_{n-k,n}$, $F_{k,n}$ and $F_{n,r}$. Notice that all the vertices in $\{a_1, \ldots, a_n\}$, $\{b_1, \ldots, b_n\}$ and X are labeled ξ_1, ξ_2 and ξ_3 respectively. From here we can easily add $F_{n-k,n}$, $F_{k,n}$ and $F_{n,r}$ with root vertices $\{a_1, \ldots, a_n\}$, $\{b_1, \ldots, b_n\}$ and X respectively by using working labels. This concludes the description for the expression tree for H. □

To conclude the proof of the first part of the Theorem 2, it remains to note that H has $4(n + m + r) \leq 4(n + m + n^2)$ vertices, and therefore if we could solve the EDGE DOMINATING SET in time $f(k)|V(H)|^{o(t)}$ where $t = \mathbf{cwd}(H)$ then same would hold for the RED-BLUE EXACT SATURATED PLANAR CDS with only difference that t is the tree-width.

Algorithmic upper bound for Edge Dominating Set: Now we give an algorithmic upper bound for the EDGE DOMINATING SET problem parameterized by the clique-width, that is, give an algorithm running in time $n^{O(t)}$ for EDGE DOMINATING SET on graphs of clique-width at most t. As usually is the case with algorithms over graphs of bounded clique-width, the algorithm is based on a dynamic programming over the expression tree of the input graph.

Let G be a graph with n vertices and m edges, and let T be an expression tree for G of the width t. For a node X of T, denote by G_X the t-graph associated with this node, and let $U_1(X), \ldots, U_t(X)$ be sets of vertices of G_X labeled $1, \ldots, t$ respectively. The table of data for the node X stores vectors $(s_1, \ldots, s_t, r_1, \ldots, r_t, l)$ of non negative integers with the following properties:

- $s_i + r_i \leq |U_i(X)|$ for $1 \leq i \leq t$;
- $l \leq |E(G_X)|$;
- there is a set of edges $S \subseteq E(G_X)$ such that s_i vertices of $U_i(X)$ are adjacent to the edges of S for $1 \leq i \leq t$, and $|S| \leq l$;

- it is possible to attach r_i pendent edges to the vertices of $U_i(X)$ for $1 \leq i \leq t$ in such a way that these edges dominate all edges of G_X undominated by S.

The size of this table is at most $(n+1)^{2t} \cdot m$. If X is the root node of T (that is $G = G_X$) then the size of the minimum edge dominating set is the minimum value of l for which the table for X contains an entry with the value of the parameter being l and $r_1 = \ldots = r_t = 0$.

Now we give the details of how we make our tables and how do we update it.

Introduce Node: Tables for introduce nodes of T are constructed in a straightforward way.

Relabel Node: Let X be a relabel node $\rho_{i \to j}$, and let Y be the child of X. Then the table for X contains a vector $(s_1, \ldots, s_t, r_1, \ldots, r_s, l)$ if and only if $s_i = 0$, $r_i = 0$ and the table for Y contains the entry $(s'_1, \ldots, s'_t, r'_1, \ldots, r'_t, l)$ such that $s'_p = s_p$ and $r_p = r'_p$ for $1 \leq p \leq t$, $t \neq i, j$, and $s_j = s'_i + s'_j$, $r_j = r'_i + r'_j$.

Union Node: Let X be a union node with children Y and Z. In this case the table for X contains a vector $(s_1, \ldots, s_t, r_1, \ldots, r_t, l)$ if and only if the tables for Y and Z have vectors $(s'_1, \ldots, s'_t, r'_1, \ldots, r'_t, l')$ and $(s''_1, \ldots, s''_t, r''_1, \ldots, r''_t, l'')$ respectively such that $s'_i + s''_i = s_i$ and $r'_i + r''_i = r_i$ for $1 \leq i \leq t$, and $l' + l'' \leq l$.

Join Node: Finally, suppose that X is a join node $\eta_{i,j}$ with the child Y. It can be noted that the table for X has a vector $(s_1, \ldots, s_t, r_1, \ldots, r_t, l)$ if and only if the table for Y includes a vector $(s'_1, \ldots, s'_t, r'_1, \ldots, r'_t, l')$ such that

- $s_p = s'_p$ and $r_p = r'_p$ for $1 \leq p \leq t$, $p \neq i, j$;
- $s_i + r_i = s'_i + r'_i$, $s_j + r_j = s'_i + r'_j$;
- $s'_i \leq s_i$, $s'_j \leq s_j$;
- either $s'_i + r'_i = |U_i(Y)|$ or $s'_j + r'_j = |U_j(Y)|$;
- $l' + \max\{s_i - s'_i, s_j - s'_j\} \leq l$.

Correctness of the algorithm follows from the description of the algorithm, and it runs in time $O(t^{O(1)} n^{4t+O(1)})$. This implies that the EDGE DOMINATIN SET problem can be solved in time $n^{O(t)}$, where t is the clique-width of the input graph. This concludes the proof of the Theorem 2.

6.5 Proof of Theorem 3

To prove our result we give a linear FPT-reduction from the RED-BLUE PLANAR CDS problem to the HAMILTONIAN CYCLE problem. The proof is organized as follows: we first give a construction, then prove its correctness and finally argue on the clique-width of the transformed instance. We call a graph *hamiltonian* if it contains a hamiltonian cycle. We start with descriptions of our auxiliary gadgets.

Auxiliary gadgets: We denote by L_1, the graph with the vertex set $\{x, y, z, a, b, c, d\}$ and the edge set $\{xa, ab, bc, cd, dy, bz, cz\}$. Let P_1 be the path $xabzcdy$, and $P_2 = xabcdy$. (See Fig. 3.)

We abstract a property of this graph in the following lemma.

Lemma 9. *Let G be a Hamiltonian graph such that $G[V']$ is isomorphic to L_1. Furthermore, if all edges in $E(G) \setminus E(G[V'])$ incident to V' are incident to the copies of the vertices x, y, and z in V', then every Hamiltonian cycle in G either includes the path P_1, or the path P_2 as a segment.*

Our second auxiliary gadget is the graph L_2. This graph has $\{x, y, z, s, t, a, b, c, d, e, f, g, h\}$ as its vertex set. We first include following $\{xa, ab, bz, cz, cd, dy, se, ef, fb, ch, hg, gt\}$ in its edge set. Then x, y-path of length 10, $xw_1 \cdots w_9 y$, is added, and edges $fw_3, w_1w_6, w_4w_9, w_7h$ are included in the set of edges. Let $P = xabzcdy$, $R_1 = sefbaxw_1w_2 \ldots w_9ydchgt$, and $R_2 = sefw_3w_2w_1w_6w_5w_4w_9w_8w_7hgt$. (See Fig. 3.) This graph has the following property.

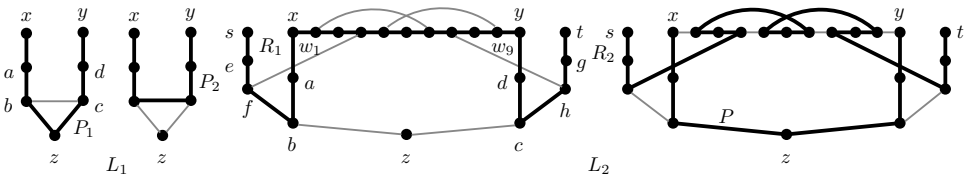

Figure 3: Graphs L_1 and L_2. Paths P_1, P_2, R_1, R_2 and P are shown by thick lines

Lemma 10. *Let G be a Hamiltonian graph such that $G[V']$ is isomorphic to L_2. Furthermore, if all edges in $E(G) \setminus E(G[V'])$ incident to V' are incident to the copies of the vertices x, y, z, s, t in V', then every Hamiltonian cycle in G includes either the path R_1, or two paths P and R_2 as segments.*

The lemma easily follows from the presence of degree 2 vertices in the graph L_2, since for any such vertex, the vertex and the adjacent vertices have to belong to one segment of a Hamiltonian path.

Final Reduction: Now we describe our reduction. Let (G, c) be red-blue capacitated graph with $R = \{u_1, \ldots, u_n\}$ being the set of red vertices and $B = \{v_1, \ldots, v_r\}$ being the set of blue vertices and k be a positive integer. Each red vertex u_i is replaced by two vertices a_i, b_i, the vertices a_i and b_i are joined by $c(u_i) + 1$ paths of length two. Let C_i denote the set of middle vertices of these paths, and $X_i = C_i \cup \{a_i, b_i\}$. Each edge $u_i v_j \in E(G)$, is replaced by a copy L_2^{ij} of L_2 with $z = v_j$ and vertices x and y are made adjacent to all the vertices of C_i. The vertices corresponding to s and t are called s_{ij} and t_{ij} in L_2^{ij}. Furthermore, let x_{ij} and y_{ij} denote the vertices corresponding to x and y in L_2^{ij}. The paths corresponding to P, R_1 and R_2 are called P^{ij}, R_1^{ij} and R_2^{ij} respectively in L_2^{ij}. Denote the obtained graph by $G'(c)$. (See Fig. 4 for an illustration.)

In the next step we add two vertices g and h which are joined by $\sum_{i=1}^{n}(c(v_i) + 3) + m + 1$ paths of length two. Let Y be the set of middle vertices of these paths. All vertices s_{ij} and t_{ij} are joined by edges with all vertices of Y. For every vertex w such that $w \in X_i$ (recall $X_i = C_i \cup \{a_i, b_i\}$), $i \in \{1, \ldots, n\}$, a copy L_1^w of L_1 with $z = w$ is attached and the vertices x, y of this gadget are joined to all vertices of Y. We let x_w and y_w denote the vertices corresponding to x and y in L_1^w. Similarly P_1^w and P_2^w denotes paths in L_1^w corresponding to P_1 and P_2 respectively.

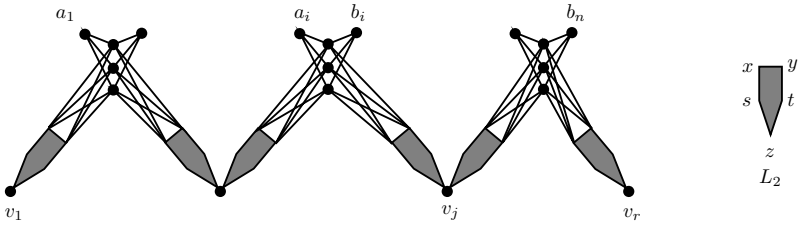

Figure 4: Graph $G'(c)$

Finally we add $k + 1$ vertices, namely $\{p_1, \ldots, p_{k+1}\}$, and make them adjacent to all the vertices $\{a_i, b_i : 1 \leq i \leq n\}$ and to g and h. Let this resultant graph be H. The construction of H can easily be done in time polynomial in n and m.

Lemma 11. *A graph (G, c) has a capacitated dominating set of size at most k if and only if H has a Hamiltonian cycle.*

Proof. Let S be a capacitated dominating set of size at most k in (G, c) with the corresponding dominating mapping f. Without loss of a generality we assume that $|S| = k$ and $S = \{u_1, \ldots, u_k\}$.

The Hamiltonian cycle we are trying to construct is naturally divided into $k+1$ parts by the vertices $\{p_1, \ldots, p_{k+1}\}$. We construct the Hamiltonian cycle starting from the vertex p_1. Assume that the part of the cycle up to the vertex p_i is already constructed. We show how to construct the part from p_i to p_{i+1}. We include the edge $p_i a_i$ in it. Let $J = \{j \in \{1, \ldots, x\} \colon f(v_j) = u_i\}$. If $J = \emptyset$ then a_i is joined with b_i by a path of length two which goes through one vertex of C_i. Otherwise, for all gadgets L_2^{ij} where $j \in J$, the paths P^{ij} are included to the cycle as segments, and endpoints of these paths are joined consecutively by paths of length two through vertices of C_i with a_i and b_i (that is, a_i is joined with one endpoint of the first path through a vertex of C_i, another endpoint of this path is joined the endpoint of the second path through another vertex of C_i and so on, the remaining endpoint of the last path is joined with b_i). Since $|J| \leq c(u_i)$ and $|C_i| = c(u_i)$, we can always find vertices in C_i for this construction. Finally we include the edge $b_i p_{i+1}$ to the cycle.

When the vertex p_{k+1} is reached we move to the set Y. Note that at this stage all vertices $\{v_1, \ldots, v_r\}$ are already included in the cycle. We start by including the edge $p_{k+1}g$. We will add following segments to the cycle and connect them appropriately.

- For every L_2^{ij}, the path R_1^{ij} is added to the cycle if P^{ij} was not included to it, else the path R_2^{ij} is added. Note that m such paths are included to the cycle.
- For every vertex w such that $w \in X_i$ for some $i \in \{1, \ldots, n\}$, the path P_2^r is included in the cycle if w is already included in the constructed part of the cycle, else the path P_1^w is added. Clearly, we add $\sum_{i=1}^{n}(c(v_i)+3)$ paths.

Finally the total number of paths we will add is $\sum_{i=1}^{n}(c(v_i)+3) + m = |Y| - 1$. We add the segments of the paths mentioned with the help of vertices in Y, in the way we added the paths P^{ij} with the help of vertices in C_i. Let the end points of the resultant joined path be $\{q_1, q_2\}$. Notice that (a) $q_1, q_2 \in Y$ and (b) this path include all the vertices of Y. Now we add edges gq_1, $q_2 h$ and hp_1. This completes the construction of the Hamiltonian cycle.

For the reverse direction of the proof, we assume that we have been given C, a Hamiltonian cycle in H. This cycle is divided into $k+1$ segments by the vertices $p_1, \ldots, p_{k+1}$. Let $S = \big\{u_i \colon p_j a_i \in E(C), a_i p_s \notin E(C), j \neq s$, for some $j \in \{1, 2, \ldots, k+1\}\big\}$. We prove that S is a capacitated dominating set in G of cardinality at most k. We first argue about the size of S, clearly its size is upper bounded by $k+1$. To argue that it is at most k, it is enough to observe that by Lemmas 9 and 10, either $p_j g$, or $p_j h$ must be in $E(C)$ for some $j \in \{1, \ldots, k+1\}$. Now we show that S is indeed a capacitated dominating set. Our proof is based on following observations.

- By Lemma 10, every vertex v_j appears in a segment P^{ij} for some $j \in \{1, \ldots, r\}$ in C. We set the domination function $f(v_j) = u_i$ if v_j is included in the segment P^{ij} in C.
- By Lemmata 9 and 10, the endpoints of paths P^{ij} can be reached only through vertices a_i and b_i from outside of the set X_i. This implies that all paths P^{ij} which appear as segments in C for some $i \in \{1, \ldots, n\}$ are joined together and with vertices a_i and b_i into one segment of C by paths which go through vertices of C_i. It means that $u_i \in S$ and $f(B) \subseteq S$. Moreover, since $|C_i| = c(u_i) + 1$, at most $c(u_i)$ paths P^{ij} can be segments of C for each $i \in \{1, \ldots, n\}$ and therefore $|f^{-1}(u_i)| \leq c(u_i)$ for $u_i \in S$.

This concludes the proof of the lemma. □

The next lemma upper bounds the clique-width of the resultant graph H.

Lemma 12. *Let $t \geq 2$. If $\mathbf{tw}(G) \leq t$ then $\mathbf{cwd}(H) \leq 48t + 72$.*

Proof. Since $t \geq 2$, $\mathbf{tw}(I(G)) = \mathbf{tw}(G)$, and by Proposition 1, we have that $\mathbf{cwd}(I(G)) \leq c = 12t + 12$. We construct an expression tree for H using $4c + 24$ labels. Suppose that the expression tree for $I(G)$ uses t labels $\{\alpha_1, \ldots, \alpha_c\}$. To construct the expression tree for H we use following additional labels:

- Labels $\beta_1, \ldots, \beta_c$ for the vertices $v_1, \ldots, v_r$.
- Labels $\gamma_1, \ldots, \gamma_c$ and $\delta_1, \ldots, \delta_c$ for the vertices in the copies of L_2 which are adjacent to the vertices of $C_1, \ldots, C_n$ and $v_1, \ldots, v_r$.
- Labels ξ_1, ξ_2, ξ_3 for marking some vertices.
- Working labels $\zeta_1, \ldots, \zeta_{21}$.

When a vertex $w \in V(I(G))$ which corresponds to the edge $u_i v_j \in E(G)$ labeled α_p is introduced, we perform the following set of operations. We use labels $\zeta_1, \ldots, \zeta_{21}$ to create a copy of the graph $L_2 - z$ (for simplicity we use a separate label for each vertex). Then vertices x and y are relabeled by γ_p, vertices b and c (adjacent to z) by δ_p and vertices s and t by ξ_1. Remaining vertices are relabeled by ξ_3. We omit the union operations from our descriptions here and henceforth in any similar descriptions and assume that if some vertex is introduced then union is always performed.

If a vertex u_i labeled α_p is introduced, then we do the following operations. First we create the vertex a_i labeled ζ_1. Then the 6 vertices of a copy of the gadget L_1 attached to a_i labeled $\zeta_5, \ldots, \zeta_{10}$ are introduced and edges of L_1 are created by corresponding join operations. We relabel vertices x and y of this copy of L_1 by ξ_1 and remaining vertices of $L_1 - z$ are relabeled by ξ_3. Next step is to introduce b_i with label ζ_2. After this we introduce a copy of L_1 attached to b_i, relabel x and y by ξ_1 and relabel remaining vertices of $L_1 - z$ by ξ_3. Now we repeat the following $c(u_i)$ times (to create vertices of C_i with attached gadgets L_1): introduce a vertex labeled ζ_3, use labels $\zeta_5, \ldots, \zeta_{10}$ (together with the vertex labeled ζ_3) to make a copy of L_1, relabel x and y of L_1 by ξ_1, relabel vertices $L_1 - z$ by ξ_3, and relabel the vertex labeled by ζ_3 by ζ_4. Finally, vertices labeled by ζ_4 are joined with vertices labeled ζ_1 and ζ_2, the vertices a_i and b_i are relabeled by ξ_2, and vertices labeled by ζ_4 are relabeled by α_p.

When a vertex v_j labeled α_p is introduced, the vertex labeled β_p is introduced.

If in the expression tree of $I(G)$, we have join operation between two labels say α_p and α_q then we simulate this by applying join operations between α_p and γ_q, δ_p and β_q, α_q and γ_p, δ_q and β_p. The relabel operation in the expression tree of G, that is, relabel α_p to α_q is replaced by relabel α_p to α_q, β_p to β_q, γ_p to γ_q and relabel δ_p to δ_q. Union operations in the expression tree are done as before.

Now we construct vertices g and h using labels ζ_1 and ζ_2. Then $\sum_{i=1}^{n}(c(v_i)+3)+m+1$ vertices of Y labeled ζ_3 are introduced and joined with the vertices labeled ζ_1 and ζ_2. The vertices g, h are relabeled by ξ_2. Notice that all the vertices which have to be joined with vertices of Y are labeled by ξ_1. So, we join vertices labeled by ζ_3 with vertices labeled ξ_1. It remains to construct vertices $p_1, \ldots, p_{k+1}$ and connect them with the already constructed part of H. To do this we introduce $k+1$ vertices labeled ζ_4, and join them with vertices labeled ξ_2. □

Lemmata 11, 12 and Proposition 2 together imply the Theorem 3.