

# Tight Complexity Bounds for FPT Subgraph Problems Parameterized by Clique-width

Hajo Broersma, Petr A. Golovach, and Viresh Patel

School of Engineering and Computing Sciences, Durham University,  
Science Laboratories, South Road, Durham DH1 3LE, UK  
{hajo.broersma, petr.golovach, viresh.patel}@durham.ac.uk

**Abstract.** We give tight algorithmic lower and upper bounds for some double-parameterized subgraph problems when the clique-width of the input graph is one of the parameters. Let  $G$  be an arbitrary input graph on  $n$  vertices with clique-width at most  $w$ . We prove the following results.

- The DENSE (SPARSE)  $k$ -SUBGRAPH problem, which asks whether there exists an induced subgraph of  $G$  with  $k$  vertices and at least  $q$  edges (at most  $q$  edges, respectively), can be solved in time  $k^{O(w)} \cdot n$ , but it cannot be solved in time  $2^{o(w \log k)} \cdot n^{O(1)}$  unless the Exponential Time Hypothesis (ETH) fails.
- The  $d$ -REGULAR INDUCED SUBGRAPH problem, which asks whether there exists a  $d$ -regular induced subgraph of  $G$ , and the MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem, which asks whether there exists a subgraph of  $G$  with  $k$  vertices and minimum degree at least  $d$ , can be solved in time  $d^{O(w)} \cdot n$ , but they cannot be solved in time  $2^{o(w \log d)} \cdot n^{O(1)}$  unless ETH fails.

## 1 Introduction

The notion of clique-width introduced by Courcelle and Olariu [14] (we refer the reader to the survey [24] for further information on different width parameters) has now become one of the fundamental parameters in Graph Algorithms. Many problems which are hard on general graphs can be solved efficiently when the input is restricted to graphs of bounded clique-width. The meta-theorem of Courcelle, Makowsky, and Rotics [13] states that all problems expressible in  $MS_1$ -logic are *fixed parameter tractable* (FPT), when parameterized by the clique-width of the input graph (see the books of Downey and Fellows [18] and Flum and Grohe [21] for a detailed treatment of parameterized complexity). In other words, this theorem shows that any problem expressible in  $MS_1$ -logic can be solved for graphs of clique-width at most  $w$  in time  $f(w) \cdot |I|^{O(1)}$ , where  $|I|$  is the size of the input and  $f$  is a computable function depending on the parameter  $w$  only. Here, the superexponential function  $f$  is defined by a logic formula, and it grows very fast.

The basic method for constructing algorithms for graphs of bounded clique-width is to use dynamic programming along an expression tree (the definition is given in Section 2). Computing clique-width is an NP-hard problem [20], but it can be approximated and a corresponding expression tree can be constructed

in FPT-time [23, 30]. In our paper it is always assumed that an expression tree is given. In this case dynamic programming algorithms can be relatively efficient: usually single-exponential in the clique-width. A natural question to ask is whether the running times of such algorithms are asymptotically optimal up to some reasonable complexity conjectures.

The Exponential Time Hypothesis has proved to be an effective tool for establishing tight complexity bounds for parameterized problems, but there are still not many results of this nature in the literature. The *Exponential Time Hypothesis (ETH)* [25] asserts that there does not exist an algorithm for solving 3-SAT running in time  $2^{o(n)}$  on a formula with  $n$  variables; this is equivalent to the parameterized complexity conjecture that  $\text{FPT} \neq \text{M}[1]$  [17, 21]. Chen et al. [8–10] showed that there is no algorithm for  $k$ -CLIQUE running in time  $f(k)n^{o(k)}$ , for  $n$ -vertex graphs, unless ETH fails (on the other hand it is easily seen that  $k$ -CLIQUE can be solved in time  $n^{O(k)}$ ). The lower bound on the  $k$ -CLIQUE problem can be extended to some other parameterized problems via linear FPT-reductions [9, 10]. In particular, for problems parameterized by clique-width, Fomin et al. [22] proved that MAX-CUT and EDGE DOMINATING SET cannot be solved in time  $f(w)n^{o(w)}$  on  $n$ -vertex graphs of clique-width at most  $w$ , unless ETH collapses. For FPT problems, Cai and Juedes [6] proved that the parameterized version of any MaxSNP-complete problem cannot be solved in time  $2^{o(k)} \cdot |I|^{O(1)}$  if ETH holds. Here  $k$  is the natural parameter of an MaxSNP-complete problem with the instance  $I$ , i.e. the maximized function should have a value at least  $k$ .

Lokshtanov, Marx and Saurabh [28] considered several FPT problems solvable in time  $2^{O(k \log k)} \cdot |I|^{O(1)}$  and showed that a  $2^{o(k \log k)} \cdot |I|^{O(1)}$ -time algorithm for these problems would violate ETH. To do this, they introduced special restricted versions of some basic problems like  $k$ -CLIQUE on graphs with  $k \times k$  vertices (and with some other restrictions) and proved that these problems cannot be solved in time  $2^{o(k \log k)} \cdot k^{O(1)}$  unless ETH collapses. These results open the possibility of establishing algorithmic lower bounds for natural problems. We use this approach to prove asymptotically tight bounds for some double-parameterized subgraph problems when the clique-width of the input graph is one of the parameters. These results give the first known bounds for such types of problems parameterized by clique-width.

First, we consider the DENSE  $k$ -SUBGRAPH problem (also known as the  $k$ -CLUSTER problem). This problem asks whether, given a graph  $G$  and positive integers  $k$  and  $q$ , there exists an induced subgraph of  $G$  with  $k$  vertices and at least  $q$  edges. Clearly, DENSE  $k$ -SUBGRAPH is NP-hard since it is a generalization of the  $k$ -CLIQUE problem. It remains NP-hard, even when restricted to comparability graphs, bipartite graphs and chordal graphs [12], as well as on planar graphs [26]. Polynomial algorithms were given for cographs, split graphs [12], and for graphs of bounded tree-width [26]. Considerable work has been done on approximation algorithms for this problem [3, 4, 15, 19, 27].

Next, we consider some degree-constrained subgraph problems. The objective in such problems is to find a subgraph satisfying certain lower or upper bounds on the degree of each vertex. Typically it is necessary to either check the existence

of a subgraph satisfying the degree constraints or to minimize (maximize) some parameter (usually the size of the subgraph).

The  $d$ -REGULAR SUBGRAPH problem, which asks whether a given graph contains a  $d$ -regular subgraph, has been intensively studied. We mention here only some complexity results. Chvátal et al. [11] proved that this problem is NP-complete for  $d = 3$ . It was shown that the problem with  $d = 3$  remains NP-complete for planar bipartite graphs with maximum degree four, and that when  $d \geq 3$ , it is NP-complete even for bipartite graphs with maximum degree at most  $d+1$ . Some further results were given in [7, 32–34]. We consider a variant of this problem called  $d$ -REGULAR INDUCED SUBGRAPH, where we ask whether a given graph  $G$  contains a  $d$ -regular *induced* subgraph. This variant of the problem has also been studied. In particular, the parameterized complexity of different variants of the problem was considered by Moser and Thilikos [31] and by Mathieson and Szeider [29]. Observe that, trivially,  $d$ -REGULAR INDUCED SUBGRAPH can be solved in polynomial time for  $d \leq 2$ , and it easily follows from the known hardness results for  $d$ -REGULAR SUBGRAPH that  $d$ -REGULAR INDUCED SUBGRAPH is NP-complete for any fixed  $d \geq 3$ .

In [2] Amini et al. introduced the MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem. This problem asks whether, given a graph  $G$  and positive integers  $d$  and  $k$ , there exists a subgraph of  $G$  with at most  $k$  vertices and minimum degree at least  $d$ . The parameterized complexity of the problem was considered in [2]. Some other hardness and approximation results can be found in [1].

**Our main results and the organization of the paper.** In Section 2 we give some basic definitions and some preliminary results. In Section 3 we consider the DENSE  $k$ -SUBGRAPH and SPARSE  $k$ -SUBGRAPH problems. The SPARSE  $k$ -SUBGRAPH problem is dual to DENSE  $k$ -SUBGRAPH and it asks whether, given a graph  $G$  and positive integers  $k$  and  $q$ , there exists an induced subgraph of  $G$  with  $k$  vertices and at most  $q$  edges. We prove that these problems can be solved in time  $k^{O(w)} \cdot n$  for  $n$ -vertex graphs of clique-width at most  $w$  if an expression tree of width  $w$  is given, but they cannot be solved in time  $2^{o(w \log k)} \cdot n^{O(1)}$  unless ETH fails even if an expression tree of width  $w$  is included in the input. In Section 4 we consider the  $d$ -REGULAR INDUCED SUBGRAPH and MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problems. We construct dynamic programming algorithms which solve these problems in time  $d^{O(w)} \cdot n$  for  $n$ -vertex graphs of clique-width at most  $w$  if an expression tree of width  $w$  is given, and then prove that these problems cannot be solved in time  $2^{o(w \log d)} \cdot n^{O(1)}$  unless ETH fails even if an expression tree of width  $w$  is provided. We conclude the paper with some open problems.

## 2 Definitions and Preliminary Results

**Graphs.** We 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  $v \in V(G)$ ,  $E_G(v)$

denotes the set of edges incident with  $v$ . The *degree* of a vertex  $v$  is denoted by  $d_G(v)$ . For a non-negative integer  $d$ , a graph  $G$  is called *d-regular* if all vertices of  $G$  have degree  $d$ . For a graph  $G$ , the *incidence graph* of  $G$  is the bipartite graph  $I(G)$  with 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 with  $e$  in  $G$ . We denote by  $\overline{G}$  the *complement* of a graph  $G$ , i.e. the graph with vertex set  $V(G)$  such that any two distinct vertices are adjacent in  $\overline{G}$  if and only if they are non-adjacent in  $G$ . For a set of vertices  $S \subseteq V(G)$ ,  $G[S]$  denotes the subgraph of  $G$  induced by  $S$ , and by  $G - S$  we denote the graph obtained from  $G$  by the removal of all the vertices of  $S$ , i.e. the subgraph of  $G$  induced by  $V(G) \setminus S$ .

**Clique-width.** Let  $G$  be a graph, and let  $w$  be a positive integer. A  $w$ -graph is a graph whose vertices are labeled by integers from  $\{1, 2, \dots, w\}$ . We call the  $w$ -graph consisting of exactly one vertex  $v$  labeled by some integer  $i$  from  $\{1, 2, \dots, w\}$  an initial  $w$ -graph. The *clique-width*  $\mathbf{cwd}(G)$  is the smallest integer  $w$  such that  $G$  can be constructed by means of repeated application of the following four operations: (1) *introduce*: construction of an initial  $w$ -graph with vertex  $v$  labeled by  $i$  (denoted by  $i(v)$ ), (2) *disjoint union* (denoted by  $\oplus$ ), (3) *relabel*: changing the labels of each vertex labeled  $i$  to  $j$  (denoted by  $\rho_{i \rightarrow j}$ ) and (4) *join*: joining all vertices labeled by  $i$  to 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:  $i$ ,  $\oplus$ ,  $\eta$  and  $\rho$ .
- Introduce nodes  $i(v)$  are leaves of  $T$ , and they correspond to initial  $w$ -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  $\rho_{i \rightarrow j}$  has one child and is associated with the  $w$ -graph resulting from the relabeling operation  $\rho_{i \rightarrow j}$  applied to the graph corresponding to the child.
- A join node  $\eta_{i,j}$  has one child and is associated with the  $w$ -graph resulting from the join operation  $\eta_{i,j}$  applied to 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  $\mathbf{cwd}(G) \leq w$  then it is possible to construct a rooted expression tree  $T$  of  $G$  with *width*  $w$ . Given a node  $X$  of an expression tree, the graph  $G_X$  is the graph formed by the subtree of the expression tree rooted at  $X$ .

**Parameterized reductions.** We refer the reader to the books [18, 21] for a detailed treatment of parameterized complexity. Here we only define the notion of parameterized (linear) reduction, which is the main tool for establishing our results. For parameterized problems  $A, B$ , we say that  $A$  is (uniformly many:1) FPT-reducible to  $B$  if there exist functions  $f, g : \mathbb{N} \rightarrow \mathbb{N}$ , a constant  $\alpha \in \mathbb{N}$  and an algorithm  $\Phi$  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  $g(k) = O(k)$ .

**Capacitated domination.** For our reductions we use a variant of the CAPACITATED DOMINATING SET problem. The parameterized complexity of this problem, with the tree-width of the input graph being the parameter, was considered in [5, 16].

A *red-blue capacitated graph* is a pair  $(G, c)$ , where  $G$  is a bipartite graph with a vertex bipartition into sets  $R$  and  $B$ , and  $c: R \rightarrow \mathbb{N}$  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*  $f: B \rightarrow S$  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  $c(v)$ . We say that for a vertex  $v \in S$ , vertices in the set  $f^{-1}(v)$  are *dominated by*  $v$ . The RED-BLUE CAPACITATED DOMINATING SET (or RED-BLUE CDS) problem asks whether, given a red-blue capacitated graph  $(G, c)$  and a positive integer  $k$ , 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.

The next proposition immediately follows from the results proved in [22].

**Proposition 1.** *The RED-BLUE CDS and RED-BLUE EXACT SATURATED CDS problems cannot be solved in time  $f(w) \cdot n^{o(w)}$ , where  $n$  is the number of vertices of the input graph  $G$  and  $w$  is the clique-width of the incidence graph  $I(G)$ , unless ETH fails, even if an expression tree of width  $w$  for  $I(G)$  is given.*

The proof of Proposition 1 uses the result of Chen et al. [8–10] that there is no algorithm for  $k$ -CLIQUE (finding a clique of size  $k$ ) running in time  $f(k) \cdot n^{o(k)}$  unless there exists an algorithm for solving 3-SAT running in time  $2^{o(n)}$  on a formula with  $n$  variables. Proposition 1 was proved via a linear reduction from the  $k$ -MULTI-COLORED CLIQUE problem (see [5, 22]). The  $k$ -MULTI-COLORED CLIQUE problem asks for a given  $k$ -partite graph  $G = (V_1 \cup \dots \cup V_k, E)$ , where  $V_1, \dots, V_k$  are sets of the  $k$ -partition, whether there is a  $k$ -clique in  $G$ . It should be noted that the construction of an expression tree of bounded width is part of the reduction and it is done in polynomial time. Lokshtanov, Marx and Saurabh [28] considered a special restricted variant of  $k$ -MULTI-COLORED CLIQUE called  $k \times k$ -CLIQUE. In this variant of the problem  $|V_1| = \dots = |V_k| = k$ . They proved the following.

**Proposition 2 ([28]).** *The  $k \times k$ -CLIQUE problem cannot be solved in time  $2^{o(k \log k)} \cdot n^{O(1)}$ , where  $n$  is the number of vertices of the input graph  $G$ , unless ETH fails.*

By replacing  $k$ -MULTI-COLORED CLIQUE by the  $k \times k$ -CLIQUE problem in the reductions used for the proof of Proposition 1, we obtain the following corollary.

**Corollary 1.** *The RED-BLUE CDS and RED-BLUE EXACT SATURATED CDS problems cannot be solved in time  $2^{o(w \log n)} \cdot n^{O(1)}$ , where  $n$  is the number of vertices of the input graph  $G$  and  $w$  is the clique-width of the incidence graph  $I(G)$ , unless ETH fails, even if an expression tree of width  $w$  for  $I(G)$  is given.*

### 3 Sparse and Dense $k$ -Subgraph problems

In this section we consider the DENSE  $k$ -SUBGRAPH and SPARSE  $k$ -SUBGRAPH problems. The aim of this section is the proof of the following theorem.

**Theorem 1.** *The SPARSE  $k$ -SUBGRAPH problem can be solved in time  $k^{O(w)} \cdot n$  on  $n$ -vertex graphs of clique-width at most  $w$  if an expression tree of width  $w$  is given, but it cannot be solved in time  $2^{o(w \log k)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

Clearly, SPARSE  $k$ -SUBGRAPH and DENSE  $k$ -SUBGRAPH are dual, i.e. SPARSE  $k$ -SUBGRAPH is equivalent to DENSE  $k$ -SUBGRAPH for the complement of the input graph. Since for any graph  $G$ ,  $\mathbf{c wd}(\overline{G}) \leq 2 \cdot \mathbf{c wd}(G)$  (see e.g. [14, 35]), we can immediately get the following corollary.

**Corollary 2.** *The DENSE  $k$ -SUBGRAPH problem can be solved in time  $k^{O(w)} \cdot n$  on  $n$ -vertex graphs of clique-width at most  $w$  if an expression tree of width  $w$  is given, but it cannot be solved in time  $2^{o(w \log k)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

#### 3.1 Algorithmic upper bounds for Sparse $k$ -Subgraph

We sketch a dynamic programming algorithm for solving SPARSE  $k$ -SUBGRAPH in time  $k^{O(w)} \cdot n$  on graphs of clique-width at most  $w$ . We describe what we store in the tables corresponding to the nodes in an expression tree.

Let  $G$  be a graph with  $n$  vertices and let  $T$  be an expression tree for  $G$  of width  $w$ . For a node  $X$  of  $T$ , let  $U_1(X), \dots, U_w(X)$  be the sets of vertices of  $G_X$  labeled  $1, \dots, w$ , respectively. The table of data for the node  $X$  contains entries which store a positive integer  $p \leq q$  and a vector  $(s_1, \dots, s_w)$  of non-negative integers such that  $s = s_1 + \dots + s_w \leq k$  for  $i \in \{1, \dots, w\}$ , for which  $p$  is the minimum number of edges of an induced subgraph  $H$  with  $s$  vertices such that for  $i \in \{1, \dots, w\}$ ,  $s_i = |U_i(X) \cap V(H)|$ . If  $X$  is the root node of  $T$  then  $G$  contains an induced subgraph with  $k$  vertices and at most  $q$  edges if and only if the table for  $X$  contains an entry with the parameter  $p \leq q$  and vector  $(s_1, \dots, s_w)$  such that  $s_1 + \dots + s_w = k$ .

The details how the tables are created and updated are given in Appendix A because of the space restrictions. Correctness of the algorithm follows from the description of the procedure.

Since for each  $X$ , the table for  $X$  contains at most  $(k+1)^w$  vectors and for each vector only one value of the parameter  $p$  is stored, the algorithm runs in time  $k^{O(w)} \cdot n$ . This proves that SPARSE  $k$ -SUBGRAPH can be solved in time  $k^{O(w)} \cdot n$  on graphs of clique-width at most  $w$ .

### 3.2 Lower bounds

To prove our lower bounds we give a reduction from the RED-BLUE CDS problem parameterized by the clique-width of the incidence graph of the input graph.

**Construction.** Let  $(G, c, k)$  be an instance of RED-BLUE CDS with  $R = \{u_1, \dots, u_n\}$  being the set of red vertices and  $B = \{v_1, \dots, v_r\}$  being the set of blue vertices. Let  $m$  be the number of edges of  $G$ . We assume without loss of generality that  $G$  has no isolated vertices. Hence,  $m \geq n, r$ .

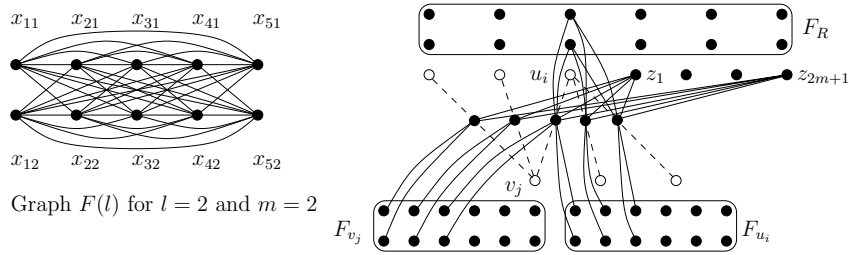
First, we construct the auxiliary gadget  $F(l)$  for a positive integer  $l$ .

**Auxiliary gadget  $F(l)$ :** Construct an  $l+m+1$ -partite graph  $K_{2, \dots, 2}$  and denote by  $x_{i1}, x_{i2}$  the vertices of the  $i$ -th set of the partition (see Figure 1).

**Reduction:** Now we describe our reduction.

1. A copy of a gadget  $F(k)$  is constructed. Denote this graph by  $F_R$  and let  $V(F_R) = \{x_{i1}^R, x_{i2}^R | 1 \leq i \leq k+m+1\}$ .
2. For each  $i \in \{1, \dots, n\}$ , a copy of a gadget  $F(c(u_i))$  is created. Denote this graph by  $F_{u_i}$  and let  $V(F_{u_i}) = \{x_{j1}^{u_i}, x_{j2}^{u_i} | 1 \leq j \leq c(u_i) + m + 1\}$ .
3. For each  $i \in \{1, \dots, r\}$ , a copy of a gadget  $F(1)$  is created. Denote this graph by  $F_{v_i}$  and let  $V(F_{v_i}) = \{x_{j1}^{v_i}, x_{j2}^{v_i} | 1 \leq j \leq m + 2\}$ .
4. For each  $e \in E(G)$ , the vertex  $w_e$  is constructed.
5. For each  $i \in \{1, \dots, n\}$ , let  $\{e_1, \dots, e_{d_i}\} = E(u_i)$  for  $d_i = d_G(u_i)$ . We consider the vertices  $w_{e_1}, \dots, w_{e_{d_i}}$ ; these vertices are joined by edges to the vertices  $x_{i1}^R, x_{i2}^R$  of  $F_R$ , and for each  $j \in \{1, \dots, d_i\}$ ,  $w_{e_j}$  is joined by edges to the vertices  $x_{j1}^{u_i}, x_{j2}^{u_i}$  of  $F_{u_i}$ .
6. For each  $i \in \{1, \dots, r\}$ , let  $\{e_1, \dots, e_{d_i}\} = E(v_i)$  for  $d_i = d_G(v_i)$ . We consider the vertices  $w_{e_1}, \dots, w_{e_{d_i}}$  and for each  $j \in \{1, \dots, d_i\}$ ,  $w_{e_j}$  is joined by edges to the vertices  $x_{j1}^{v_i}, x_{j2}^{v_i}$  of  $F_{v_i}$ .
7. Create  $2m + 1$  vertices  $z_1, \dots, z_{2m+1}$  and join them to all vertices  $w_e$  for  $e \in E(G)$ .

Denote the obtained graph by  $H$  (see Figure 1).



**Fig. 1.** Construction of  $H$ .

Due the space restrictions the proof of the following lemma is given in Appendix B.

**Lemma 1.** *The red-blue graph  $G$  has a capacitated dominating set of size at most  $k$  if and only if  $H$  contains an induced subgraph with  $2(m+1)(n+r+1) + 2m+1+r$  vertices and at most  $2m(m+1)(n+r+1) + r(2m+1)$  edges.*

We prove an upper bound for the clique-width of  $H$  as a linear function in the clique-width of the incidence graph  $I(G)$  of  $G$  (the proof is given in Appendix C).

**Lemma 2.** *We have  $\text{cwd}(H) \leq 9 \cdot \text{cwd}(I(G)) + 1$  and an expression tree of width at most  $9 \cdot \text{cwd}(I(G)) + 1$  for  $H$  can be constructed in polynomial time given an expression tree of width  $\text{cwd}(I(G))$  for  $I(G)$ .*

To complete the proof of Theorem 1, notice that the number of vertices of  $H$  and the parameter  $k$  are polynomial in  $n+r$ . Therefore,  $\log k$  is linear in  $\log(n+k)$ , and if we could solve SPARSE  $k$ -SUBGRAPH in time  $2^{o(\text{cwd}(H) \log k)} \cdot |V(H)|^{O(1)}$  then RED-BLUE CDS could be solved in time  $2^{o(\text{cwd}(I(G)) \log |V(G)|)} \cdot |V(G)|^{O(1)}$ . By Corollary 1, it cannot be done unless ETH fails.

## 4 Degree-constrained subgraph problems

The first aim of this section is the proof of the following theorem.

**Theorem 2.** *The  $d$ -REGULAR INDUCED SUBGRAPH problem can be solved on  $n$ -vertex graphs of clique-width at most  $w$  in time  $d^{O(w)} \cdot n$  if an expression tree of width  $w$  is given for the input graph, but it cannot be solved in time  $2^{o(w \log d)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

*Proof.* The algorithmic upper bounds are proved by constructing a dynamic programming algorithm for solving  $d$ -REGULAR INDUCED SUBGRAPH in time  $d^{O(w)} \cdot n$  on graphs of clique-width at most  $w$ . Details of the algorithm are given in Appendix D. To prove our complexity lower bound, we give a reduction from the RED-BLUE EXACT SATURATED CDS problem, parameterized by the clique-width of the incidence graph of the input graph, to the  $d$ -REGULAR INDUCED SUBGRAPH problem. The proof is organized as follows: we first give a construction, then prove its correctness and finally bound the clique-width of the transformed instance.

**Construction.** Let  $(G, c, k)$  be an instance of RED-BLUE EXACT SATURATED CDS with  $R = \{u_1, \dots, u_n\}$  being the set of red vertices and  $B = \{v_1, \dots, v_r\}$  being the set of blue vertices. Let  $d = n+r+1$  if  $n+r$  is even and let  $d = n+r+2$  otherwise; notice that  $d$  is odd. We need an auxiliary gadget.

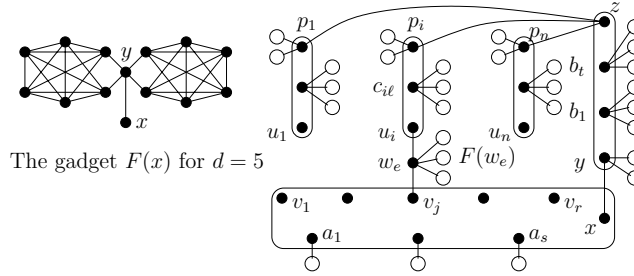
**Auxiliary gadget  $F(x)$ :** Let  $x$  be a vertex. We construct  $\frac{d-1}{2}$  copies of  $K_{d+1}$ , subdivide one edge of each copy, and glue (identify) all these vertices of degree two into one vertex  $y$ . Finally we join  $x$  and  $y$  by an edge. We are going to attach gadgets  $F(x)$  to other parts of our construction through the vertex  $x$ . This vertex is called the *root* of  $F(x)$ . The gadget  $F(x)$  for  $d = 5$  is illustrated in Figure 2.

**Reduction:** Now we describe our reduction. Let  $s = d - r - 1$  and  $t = d - k - 1$ .



1. Vertices  $u_1, \dots, u_n$  are created.
2. A clique of size  $r$  with vertices  $v_1, \dots, v_r$  is constructed.
3. For each edge  $e = u_i v_j$  of  $G$ , a vertex  $w_e$  is added, joined by edges to  $u_i$  and  $v_j$ , and  $d - 2$  copies of  $F(w_e)$  are constructed.
4. A clique of size  $s$  with vertices  $a_1, \dots, a_s$  is created, all vertices  $a_i$  are joined to vertices  $v_1, \dots, v_r$ , and for each  $i \in \{1, \dots, s\}$ , a copy of  $F(a_i)$  is added.
5. A vertex  $x$  is introduced and joined by edges to  $v_1, \dots, v_r$  and  $a_1, \dots, a_s$ .
6. A vertex  $y$  is added and joined by an edge to  $x$ , and  $k - 1$  copies of  $F(y)$  are added.
7. A clique of size  $t$  with vertices  $b_1, \dots, b_t$  is constructed, the vertex  $y$  is joined by edges to all vertices of the clique, and for each  $j \in \{1, \dots, t\}$ ,  $k$  copies of  $F(b_j)$  are added.
8. A vertex  $z$  is introduced and joined by edges to vertices  $y$  and  $b_1, \dots, b_t$ .
9. For each  $i \in \{1, \dots, n\}$ , we let  $l_i = d - c(u_i) - 1$  and do the following:
  - Add a vertex  $p_i$ , join it to  $z$  by an edge, and construct  $c(u_i) - 1$  copies of  $F(p_i)$ .
  - Construct a clique of size  $l_i$  with vertices  $c_{i1}, \dots, c_{il_i}$ , join them to the vertex  $p_i$  by edges, and for each  $j \in \{1, \dots, l_i\}$ , introduce  $c(u_i)$  copies of  $F(c_{ij})$ .
  - Join the vertex  $u_i$  to the vertices  $p_i$  and  $c_{i1}, \dots, c_{il_i}$  by edges.

Denote the obtained graph by  $H$ . The construction of  $H$  is illustrated in Figure 2.



**Fig. 2.** Construction of  $H$ .

The proof of the following lemma is given in Appendix E.

**Lemma 3.** *The red-blue graph  $G$  has an exact saturated capacitated dominating set of size  $k$  if and only if  $H$  contains an induced  $d$ -regular subgraph.*

Now we show that the clique-width of  $H$  is bounded from above by a linear function in the clique-width of the incidence graph  $I(G)$  of  $G$  (the proof is in Appendix F).

**Lemma 4.** *We have that  $\text{cwd}(H) \leq 3 \cdot \text{cwd}(I(G)) + 6$  and an expression tree of width at most  $3 \cdot \text{cwd}(I(G)) + 6$  for  $H$  can be constructed in polynomial time assuming we are given an expression tree of width  $\text{cwd}(I(G))$  for  $I(G)$ .*

To conclude this part of the proof of Theorem 2, we observe that the number of vertices of  $H$  and the parameter  $d$  are polynomial in  $n + r$ , and therefore if we could solve  $d$ -REGULAR INDUCED SUBGRAPH in time  $2^{o(\text{cwd}(H) \log d)} \cdot |V(H)|^{O(1)}$  then the RED-BLUE EXACT SATURATED CDS could be solved in time  $2^{o(\text{cwd}(I(G)) \log |V(G)|)} \cdot |V(G)|^{O(1)}$ . By Corollary 1, this cannot be done unless ETH fails.  $\square$

In the  $d$ -REGULAR INDUCED SUBGRAPH problem we ask about the existence of a  $d$ -regular induced subgraph for a given graph. It is possible to get similar results for some variants of this problem. The MINIMUM  $d$ -REGULAR INDUCED SUBGRAPH problem and the MAXIMUM  $d$ -REGULAR INDUCED SUBGRAPH problem are respectively the problems of finding a  $d$ -regular induced subgraph of minimum and maximum size. For the COUNTING  $d$ -REGULAR INDUCED SUBGRAPH problem, we are interested in the number of induced  $d$ -regular subgraphs of the input graph. Using Theorem 2 we get the following corollary (the proof is given in Appendix G).

**Corollary 3.** *The MINIMUM  $d$ -REGULAR INDUCED SUBGRAPH, MAXIMUM  $d$ -REGULAR INDUCED SUBGRAPH and COUNTING  $d$ -REGULAR INDUCED SUBGRAPH problems can be solved on  $n$ -vertex graphs of clique-width at most  $w$  in time  $d^{O(w)} \cdot n$  if an expression tree of width  $w$  is given, but they cannot be solved in time  $2^{o(w \log n)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

We conclude this section by considering the MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem. The proof of the following theorem is given in Appendix H.

**Theorem 3.** *The MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem can be solved on  $n$ -vertex graphs of clique-width at most  $w$  in time  $d^{O(w)} \cdot n$  if an expression tree of width  $w$  is given, but it cannot be solved in time  $2^{o(w \log d)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

## 5 Conclusion

We established tight algorithmic lower and upper bounds for some double-parameterized subgraph problems when the clique-width of the input graph is one of the parameters. We believe that similar bounds could be given for other problems.

Another interesting task is to consider problems parameterized by other width-parameters. Throughout the paper, in all our results we assumed that an expression tree of the given width is part of the input. This is crucial, since — unlike the case of tree-width — to date we are unaware of an efficient (FPT or polynomial) algorithm for computing an expression tree with a constant factor approximation of the clique-width. The algorithm given by Oum and Seymour in [30] provides a constant factor approximation for another graph parameter —

*rank-width* [24, 30]. Hence, it is natural to ask whether it is possible to establish tight algorithmic bounds for DENSE  $k$ -SUBGRAPH,  $d$ -REGULAR INDUCED SUBGRAPH and MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  parameterized by the rank-width of the input graph. Also it would be interesting to consider problems parameterized by the tree-width. For example, it can be shown that  $d$ -REGULAR INDUCED SUBGRAPH and MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  can be solved in time  $d^{O(t)} \cdot n$  for  $n$ -vertex graphs of tree-width at most  $t$ . Is this bound asymptotically tight?

## References

1. O. AMINI, D. PELEG, S. PÉRENNES, I. SAU, AND S. SAURABH, *Degree-constrained subgraph problems: Hardness and approximation results*, in WAOA, vol. 5426 of Lecture Notes in Computer Science, Springer, 2008, pp. 29–42.
2. O. AMINI, I. SAU, AND S. SAURABH, *Parameterized complexity of the smallest degree-constrained subgraph problem*, in IWPEC, vol. 5018 of Lecture Notes in Computer Science, Springer, 2008, pp. 13–29.
3. S. ARORA, D. R. KARGER, AND M. KARPINSKI, *Polynomial time approximation schemes for dense instances of  $p$ -hard problems*, in STOC, ACM, 1995, pp. 284–293.
4. Y. ASAHIRO, K. IWAMA, H. TAMAKI, AND T. TOKUYAMA, *Greedily finding a dense subgraph*, in SWAT, vol. 1097 of Lecture Notes in Computer Science, Springer, 1996, pp. 136–148.
5. H. L. BODLAENDER, D. LOKSHTANOV, AND E. PENNINKX, *Planar capacitated dominating set is  $W[1]$ -hard*, in IWPEC, vol. 5917 of Lecture Notes in Computer Science, Springer, 2009, pp. 50–60.
6. L. CAI AND D. W. JUEDES, *On the existence of subexponential parameterized algorithms*, J. Comput. Syst. Sci., 67 (2003), pp. 789–807.
7. F. CHEAH AND D. G. CORNEIL, *The complexity of regular subgraph recognition*, Discrete Applied Mathematics, 27 (1990), pp. 59–68.
8. J. CHEN, B. CHOR, M. FELLOWS, X. HUANG, D. JUEDES, I. A. KANJ, AND G. XIA, *Tight lower bounds for certain parameterized NP-hard problems*, Information and Computation, 201 (2005), pp. 216–231.
9. 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.
10. ———, *Strong computational lower bounds via parameterized complexity*, J. Comput. Syst. Sci., 72 (2006), pp. 1346–1367.
11. V. CHVÁTAL, H. FLEISCHNER, J. SHEEHAN, AND C. THOMASSEN, *Three-regular subgraphs of four-regular graphs*, J. Graph Theory, 3 (1979), pp. 371–386.
12. D. G. CORNEIL AND Y. PERL, *Clustering and domination in perfect graphs*, Discrete Appl. Math., 9 (1984), pp. 27–39.
13. 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.
14. B. COURCELLE AND S. OLARIU, *Upper bounds to the clique width of graphs*, Discrete Appl. Math., 101 (2000), pp. 77–114.
15. E. D. DEMAINE, M. T. HAJIAGHAYI, AND K. ICHI KAWARABAYASHI, *Algorithmic graph minor theory: Decomposition, approximation, and coloring*, in FOCS, IEEE Computer Society, 2005, pp. 637–646.

16. M. DOM, D. LOKSHTANOV, S. SAURABH, AND Y. VILLANGER, *Capacitated domination and covering: A parameterized perspective*, in IWPEC'08, vol. 5018 of Lecture Notes in Computer Science, Springer, 2008, pp. 78–90.
17. 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).
18. R. G. DOWNEY AND M. R. FELLOWS, *Parameterized complexity*, Monographs in Computer Science, Springer-Verlag, New York, 1999.
19. U. FEIGE, D. PELEG, AND G. KORTSARZ, *The dense  $k$ -subgraph problem*, Algorithmica, 29 (2001), pp. 410–421.
20. 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.
21. J. FLUM AND M. GROHE, *Parameterized complexity theory*, Texts in Theoretical Computer Science. An EATCS Series, Springer-Verlag, Berlin, 2006.
22. F. V. FOMIN, P. A. GOLOVACH, D. LOKSHTANOV, AND S. SAURABH, *Algorithmic lower bounds for problems parameterized by clique-width*, in SODA, 2010, pp. 493–502.
23. P. HLINENÝ AND S.-IL OUM, *Finding branch-decompositions and rank-decompositions*, SIAM J. Comput., 38 (2008), pp. 1012–1032.
24. P. HLINENÝ, S.-IL OUM, D. SEESE, AND G. GOTTLÖB, *Width parameters beyond tree-width and their applications*, Comput. J., 51 (2008), pp. 326–362.
25. R. IMPAGLIAZZO, R. PATURI, AND F. ZANE, *Which problems have strongly exponential complexity?*, J. Comput. Syst. Sci., 63 (2001), pp. 512–530.
26. J. M. KEIL AND T. B. BRECHT, *The complexity of clustering in planar graphs*, J. Combin. Math. Combin. Comput., 9 (1991), pp. 155–159.
27. G. KORTSARZ AND D. PELEG, *On choosing a dense subgraph (extended abstract)*, in FOCS, IEEE, 1993, pp. 692–701.
28. D. LOKSHTANOV, D. MARX, AND S. SAURABH, *Slightly superexponential parameterized problems*, in SODA, 2011.
29. L. MATHIESON AND S. SZEIDER, *The parameterized complexity of regular subgraph problems and generalizations*, in CATS, vol. 77 of CRPIT, Australian Computer Society, 2008, pp. 79–86.
30. S.-IL OUM AND P. SEYMOUR, *Approximating clique-width and branch-width*, J. Combin. Theory Ser. B, 96 (2006), pp. 514–528.
31. H. MOSER AND D. M. THILIKOS, *Parameterized complexity of finding regular induced subgraphs*, J. Discrete Algorithms, 7 (2009), pp. 181–190.
32. I. A. STEWART, *Deciding whether a planar graph has a cubic subgraph is np-complete*, Discrete Mathematics, 126 (1994), pp. 349–357.
33. ———, *Finding regular subgraphs in both arbitrary and planar graphs*, Discrete Applied Mathematics, 68 (1996), pp. 223–235.
34. ———, *On locating cubic subgraphs in bounded-degree connected bipartite graphs*, Discrete Mathematics, 163 (1997), pp. 319–324.
35. E. WANKE,  *$k$ -NLC graphs and polynomial algorithms*, Discrete Appl. Math., 54 (1994), pp. 251–266.

## Appendix

### A. Algorithmic upper bounds for Sparse $k$ -Subgraph

Recall that we store in the tables corresponding to the nodes in an expression tree the following information. Let  $G$  be a graph with  $n$  vertices and let  $T$  be an expression tree for  $G$  of width  $w$ . Let  $U_1(X), \dots, U_w(X)$  be the sets of vertices of  $G_X$  labeled  $1, \dots, w$ , respectively. The table of data for the node  $X$  contains entries which store a positive integer  $p \leq q$  and a vector  $(s_1, \dots, s_w)$  of non-negative integers such that  $s = s_1 + \dots + s_w \leq k$ , for which  $p$  is the minimum number of edges of an induced subgraph  $H$  with  $s$  vertices such that for  $i \in \{1, \dots, w\}$ ,  $s_i = |U_i(X) \cap V(H)|$ . If  $X$  is the root node of  $T$  then  $G$  contains an induced subgraph with  $k$  vertices and at most  $q$  edges if and only if the table for  $X$  contains an entry with the parameter  $p \leq q$  and vector  $(s_1, \dots, s_w)$  such that  $s_1 + \dots + s_w = k$ .

We create and update the tables as follows.

**Introduce Node:** Tables for introduce nodes of  $T$  are constructed in a straightforward manner. Suppose that  $X = i(v)$  for  $v \in V(G)$  and  $i \in \{1, \dots, w\}$ .

Then the table of data for the node  $X$  contains two entries with  $p = 0$  and the vectors  $(s_1, \dots, s_w)$  such that  $s_j = 0$  for  $j \in \{1, \dots, w\}$ ,  $j \neq i$ ,  $s_i = 0$  for the first entry and  $s_i = 1$  for the second.

**Relabel Node:** Suppose that  $X$  is a relabel node  $\rho_{i \rightarrow j}$ , and  $Y$  is a child of  $X$ . Then the table for  $X$  contains an entry with an integer  $p$  and a vector  $(s_1, \dots, s_w)$  if and only if  $s_i = 0$  and  $p$  is the minimum integer for which the table for  $Y$  contains the entry with  $p$  and the vector  $(s'_1, \dots, s'_w)$  such that

- $s_l = s'_l$  for  $l \in \{1, \dots, w\}$ ,  $l \neq i, j$ ,
- $s_j = s'_i + s'_j$ .

**Union Node:** Let  $X$  be a union node with children  $Y$  and  $Z$ . In this case the table for  $X$  contains an entry with  $p$  and a vector  $(s_1, \dots, s_w)$  if and only if  $p$  is the minimum integer for which the tables for  $Y$  and  $Z$  have the entries  $p'$ ,  $(s'_1, \dots, s'_w)$  and  $p''$ ,  $(s''_1, \dots, s''_w)$  respectively, such that

- $p = p' + p''$ ,
- $s_i = s'_i + s''_i$  for  $i \in \{1, \dots, w\}$ .

**Join Node:** Finally, suppose that  $X$  is a join node  $\eta_{i,j}$  with a child  $Y$ . It can be noted that the table for  $X$  has an entry  $p$ ,  $(s_1, \dots, s_w)$  if and only if  $p$  is the minimum integer for which the table for  $Y$  includes the entry  $p'$ ,  $(s_1, \dots, s_w)$  where  $p = p' + s_i s_j$ .

### B. Proof of Lemma 1

We need an additional technical lemma about properties of the gadget  $F(l)$ .

**Lemma 5.** *Let  $J$  be a disjoint union of  $s$  copies  $F_1, \dots, F_s$  of  $F(l_1), \dots, F(l_s)$ , respectively for  $l_1, \dots, l_s \geq 1$ , and let  $U \subseteq V(J)$ .*

- If  $|U| = 2s(m+1)$  then  $|E(G[U])| \geq 2sm(m+1)$ , and  $|E(G[U])| = 2sm(m+1)$  if and only if  $U$  contains vertices of  $m+1$  sets of the  $(l_i+m+1)$ -partition of each copy of  $F(l_i)$ .
- If  $|U| < 2s(m+1)$  then  $2sm(m+1) - |E(G[U])| \leq (2s(m+1) - |U|) \cdot 2m$ .
- If  $|U| > 2s(m+1)$  then  $|E(G[U])| - 2sm(m+1) \geq (|U| - 2s(m+1)) \cdot 2(m+1)$ .

*Proof.* For  $s = 1$ , the first claim follows immediately from the definition of  $F(l)$ . Let  $s > 1$ . Suppose that  $U \subseteq V(J)$  contains  $2s(m+1)$  vertices and the number of edges of  $J[U]$  is minimum. Let  $p_i = V(F_i) \cap U$  for  $i \in \{1, \dots, s\}$ . Since  $U$  induces a subgraph with the minimum number of edges,  $U$  contains vertices of  $\lfloor p_i/2 \rfloor$  sets of the  $(l_i+m+1)$ -partition of each  $F_i$ . Assume that there are  $i, j \in \{1, \dots, s\}$  such that  $p_i > 2(m+1)$  and  $p_j < 2(m+1)$ . Observe that there is a vertex  $u \in V(F_i) \cap U$  which is adjacent to at least  $2(m+1)$  vertices of  $U$  and there is a vertex  $v \in V(F_j) \setminus U$  which is adjacent to at most  $2m$  vertices of  $U$ . Consider the set  $U' = (U \setminus \{u\}) \cup \{v\}$ . Clearly,  $|E(J[U'])| \leq |E(J[U])| - 2(m+1) + 2m < |E(J[U])|$ ; this contradicts our choice of  $U$ . Therefore  $p_i = 2(m+1)$ .

The second claim follows from the fact that if  $|U| < 2s(m+1)$  and  $G[U]$  has the minimum number of edges then we can add  $2s(m+1) - |U|$  vertices of degree at most  $2m$ . The third claim is proved by similar arguments.  $\square$

Now we can prove Lemma 1.

**Lemma 1.** *The red-blue graph  $G$  has a capacitated dominating set of size at most  $k$  if and only if  $H$  contains an induced subgraph with  $2(m+1)(n+r+1) + 2m+1+r$  vertices and at most  $2m(m+1)(n+r+1) + r(2m+1)$  edges.*

*Proof.* Suppose that the red-blue graph  $G$  has a capacitated dominating set of size at most  $k$ . Let  $f$  be a corresponding domination mapping. We construct an induced subgraph  $Q$  of  $H$  as follows:

- Include all the vertices  $z_1, \dots, z_{2m+1}$  in the set of vertices of  $Q$ .
- For  $F_R$ , select  $m+1$  sets of the  $(k+m+1)$ -partition  $\{x_{i1}^R, x_{i2}^R\}$  such that  $u_i \notin S$  and include these vertices in  $V(Q)$ .
- For each  $u_i \in S$ , consider  $c \leq c(u_i) \leq d_G(u_i)$  edges  $e_1, \dots, e_c \in E(u_i)$  which are used for domination (i.e. for each  $e_j = u_i v_h$ ,  $f(v_h) = u_i$ ). Select  $m+1$  sets of the  $(c(u_i)+m+1)$ -partition  $\{x_{j1}^{u_i}, x_{j2}^{u_i}\}$  of  $F_{u_i}$  such that  $x_{j1}^{u_i}, x_{j2}^{u_i}$  are not adjacent to  $w_{e_1}, \dots, w_{e_c}$  and include these vertices in  $V(Q)$ .
- For each  $u_i \in R \setminus S$ , select  $m+1$  sets of the  $(c(u_i)+m+1)$ -partition  $\{x_{j1}^{u_i}, x_{j2}^{u_i}\}$  of  $F_{u_i}$  arbitrarily and include these vertices in  $V(Q)$ .
- For each  $v_i \in B$ , let  $e = v_i f(v_i) \in E(G)$ . Select  $m+1$  sets of the  $(m+2)$ -partition  $\{x_{j1}^{v_i}, x_{j2}^{v_i}\}$  of  $F_{v_i}$  such that  $x_{j1}^{v_i}, x_{j2}^{v_i}$  are not adjacent to  $w_e$  and include these vertices in  $V(Q)$ .
- For each  $v_i \in B$ , include the vertex  $w_e$  for  $e = v_i f(v_i) \in E(G)$  in  $Q$ .

It is straightforward to check that  $Q$  has  $2m+1+2(m+1)(n+r+1)+r$  vertices and  $2m(m+1)(n+r+1)+r(2m+1)$  edges.

Assume now that  $Q$  is an induced subgraph of  $H$  with  $2(m+1)(n+r+1)+2m+1+r$  vertices and the minimum number of edges, such that  $|E(Q)| \leq 2m(m+1)(n+r+1)+r(2m+1)$ .

**Claim 1.** *It can be assumed that  $z_1, \dots, z_{2m+1} \in V(Q)$ .*

*Proof (of Claim 1).* Let  $Z = \{z_1, \dots, z_{2m+1}\} \cap V(Q)$  and suppose that  $|Z| < 2m + 1$ . Let  $W = \{w_e | e \in E(G)\} \cap V(Q)$ . Assume that  $W \neq \emptyset$ . If  $|W| \geq |Z|$  then the graph obtained from  $Q$  by the replacement of the vertices of  $W$  by  $|W|$  vertices from  $\{z_1, \dots, z_{2m+1}\} \setminus Z$  contains less edges than  $Q$  and this contradicts our choice of  $Q$ . If  $|W| < |Z|$  then the graph obtained from  $Q$  by the replacement of a vertex  $u \in W$  by a vertex  $v \in \{z_1, \dots, z_{2m+1}\} \setminus Z$  has at most  $|E(Q)| - |Z| + |W| - 1 < |E(Q)|$  edges and we again have a contradiction. Therefore  $W = \emptyset$  and  $\{z_1, \dots, z_{2m+1}\}$  is an independent set in  $Q$ . Then the replacement of any  $2m + 1 - |Z|$  vertices from  $V(Q) \setminus Z$  by the vertices of  $\{z_1, \dots, z_{2m+1}\} \setminus Z$  does not increase the number of edges.  $\square$

From now on we assume that  $z_1, \dots, z_{2m+1} \in V(Q)$ .

**Claim 2.** *The set  $V(Q)$  contains  $2(m + 1)(n + r + 1)$  vertices of the gadgets  $F_R, F_{u_1}, \dots, F_{u_n}, F_{v_1}, \dots, F_{v_r}$  and  $r$  vertices from the set  $\{w_e | e \in E(G)\}$ . Moreover, each of the gadgets  $F_R, F_{u_1}, \dots, F_{u_n}, F_{v_1}, \dots, F_{v_r}$  contains exactly  $2(m + 1)$  vertices from  $m + 1$  sets of the partitions of the gadgets and the vertices from  $\{w_e | e \in E(G)\} \cap V(Q)$  are not adjacent to vertices of these gadgets in the graph  $Q$ .*

*Proof (of Claim 2).* Let  $p$  be the number of vertices of  $Q$  in the gadgets  $F_R, F_{u_1}, \dots, F_{u_n}, F_{v_1}, \dots, F_{v_r}$  and let  $q$  be the number of vertices in  $\{w_e | e \in E(G)\}$ .

Suppose that  $p < 2(m + 1)(n + r + 1)$ . By the second claim of Lemma 5,  $Q$  contains at least  $2pm - 2(n + r + 1)(m + 1)m$  edges in the graphs  $F_R, F_{u_1}, \dots, F_{u_n}, F_{v_1}, \dots, F_{v_r}$ . Each of the  $q$  vertices  $w_e$  is adjacent to the vertices  $z_1, \dots, z_{2m+1}$  in  $Q$ . Observe that  $p + q = 2(m + 1)(n + r + 1) + r$ . Hence  $Q$  contains at least  $2m(m + 1)(n + r + 1) + (2m + 1)m + (2(m + 1)(n + r + 1) - p) > 2m(m + 1)(n + r + 1) + r(2m + 1)$  edges. This contradiction proves that  $p \geq 2(m + 1)(n + r + 1)$ .

Suppose now that  $p > 2(m + 1)(n + r + 1)$ . We apply the third claim of Lemma 5 and conclude that  $Q$  has at least  $(p - 2(n + r + 1)(m + 1))2(m + 1) + 2(n + r + 1)(m + 1)m = (r - q)2(m + 1) + 2(n + r + 1)(m + 1)m$  edges. Recall that the  $q$  vertices  $w_e$  in  $Q$  are adjacent to the vertices  $z_1, \dots, z_{2m+1}$  in  $Q$ . We have that  $Q$  contains at least  $(r - q)2(m + 1) + 2(n + r + 1)(m + 1)m + q(2m + 1) = 2(n + r + 1)(m + 1)m + r(2m + 1) + r - q > 2(n + r + 1)(m + 1)m + r(2m + 1)$  edges. We conclude that  $p = 2(m + 1)(n + r + 1)$  and  $q = r$ .

Since  $p = 2(m + 1)(n + r + 1)$  and  $q = r$ ,  $Q$  has at most  $2m(m + 1)(n + r + 1)$  edges different from  $w_e z_j$ . By the first claim of Lemma 5,  $Q$  has at least  $2m(m + 1)(n + r + 1)$  edges in the gadgets  $F_R, F_{u_1}, \dots, F_{u_n}, F_{v_1}, \dots, F_{v_r}$ . Hence the vertices from  $\{w_e | e \in E(G)\} \cap V(Q)$  are not adjacent to the vertices of these gadgets in the graph  $Q$ . Also by the same claim, each of the gadgets  $F_R, F_{u_1}, \dots, F_{u_n}, F_{v_1}, \dots, F_{v_r}$  contains exactly  $2(m + 1)$  vertices of the  $m + 1$  sets of the partitions of the graph.  $\square$

Now we can complete the proof of the lemma.

The graph  $Q$  has  $r$  vertices from the set  $\{w_e | e \in E(G)\}$ . Consider a gadget  $F_{v_i}$  constructed for the vertex  $v_i \in B$ . The graph  $Q$  contains  $2(m+1)$  vertices in  $F_{v_i}$  which constitute  $m+1$  sets in the  $(m+2)$ -partition of this graph. Hence only the unique vertex  $w_e$ , for  $e \in E_G(v_i)$ , non-adjacent to the vertices of  $F_{v_i}$  in  $Q$  can be included in  $V(Q)$ . It follows that for each  $i \in \{1, \dots, r\}$ , exactly one vertex  $w_e$  is included in  $V(Q)$ . We define the mapping  $f: B \rightarrow R$  by setting  $f(v_i) = u_j$  such that  $w_e \in V(Q)$  for  $e = v_i u_j$ . Now observe that  $2(m+1)$  vertices of the graph  $F_R$  in  $m+1$  sets of the  $(k+m+1)$ -partition are in  $V(Q)$ . Thus, exactly  $k$  sets of vertices from  $\{w_e | e \in E_G(u_1)\}, \dots, \{w_e | e \in E_G(u_n)\}$  are non-adjacent to all the vertices of  $F_R$  in  $Q$ . We construct the set  $S \subset R$  by including in it vertices  $u_i \in R$  for which  $\{w_e | e \in E_G(u_i)\}$  has this property. Clearly,  $|S| = k$ . Each  $w_e \in V(Q)$  is included in one of the sets  $\{w_e | e \in E_G(u_i)\}$  for  $u_i \in S$ . It remains to observe that each set  $\{w_e | e \in E_G(u_i)\}$  can only contain at most  $c(u_i)$  vertices, namely those that are non-adjacent to the  $2(m+1)$  vertices of  $Q$  in  $F_{v_i}$ . We conclude that  $S$  is a capacitated dominating set in the red-blue graph  $G$  and  $f$  is a dominating mapping for  $S$ .  $\square$

## C. Proof of Lemma 2

**Lemma 2.** *We have  $\mathbf{cwd}(H) \leq 9 \cdot \mathbf{cwd}(I(G)) + 1$  and an expression tree of width at most  $9 \cdot \mathbf{cwd}(I(G)) + 1$  for  $H$  can be constructed in polynomial time given an expression tree of width  $\mathbf{cwd}(I(G))$  for  $I(G)$ .*

*Proof.* Let  $w = \mathbf{cwd}(I(G))$  and suppose that the expression tree for  $I(G)$  uses labels  $\{\mu_1, \dots, \mu_w\}$ . To construct an expression tree for  $H$  we use the following labels:

- labels  $\alpha_1, \dots, \alpha_w$  and  $\alpha'_1, \dots, \alpha'_w$  for the vertices of  $F_R$ ;
- labels  $\beta_1, \dots, \beta_w$  for the vertices  $w_e$ ;
- labels  $\gamma_1, \dots, \gamma_w, \gamma'_1, \dots, \gamma'_w$  and  $\delta_1, \dots, \delta_w$  for the vertices of the gadgets  $F_{u_i}$ ;
- labels  $\zeta_1, \dots, \zeta_w, \zeta'_1, \dots, \zeta'_w$  and  $\eta_1, \dots, \eta_w$  for the vertices of the gadgets  $F_{v_i}$ ;
- and
- a working label  $\lambda$ .

We construct the required expression tree for  $H$  by going over the expression tree for  $I(G)$  and making necessary changes to it.

When a vertex  $u_i \in R$  labeled by  $\mu_p$  is introduced, we perform the following set of operations. We first introduce the vertices  $x_{i1}^R, x_{i2}^R$  labeled  $\alpha_p$ . Then we construct the subgraph of  $F_{u_i}$  induced by the vertices  $x_{j1}^{u_i}, x_{j2}^{u_i}$  for  $d_G(u_i) < j \leq c(u_i) + m + 1$  by repeating  $c(u_i) + m + 1 - d_G(u_i)$  times the following: (a) introduce two vertices labeled  $\lambda$ ; (b) join the vertices labeled by  $\lambda$  to the vertices labeled by  $\delta_p$ ; (c) relabel the vertices labeled by  $\lambda$  by  $\delta_p$ .

When a vertex  $v_i \in B$  labeled by  $\mu_p$  is introduced, we construct the subgraph of  $F_{v_i}$  induced by the vertices  $x_{j1}^{v_i}, x_{j2}^{v_i}$  for  $d_G(v_i) < j \leq m + 2$  by repeating



$m + 2 - d_G(v_i)$  times the following: (a) introduce two vertices labeled  $\lambda$ ; (b) join the vertices labeled by  $\lambda$  to the vertices labeled by  $\eta_p$ ; (c) relabel the vertices labeled by  $\lambda$  by  $\eta_p$ .

When a vertex of  $I(G)$  which corresponds to an edge  $e = u_i v_j \in E(G)$  labeled  $\mu_p$  is introduced, we (a) introduce the vertex  $w_e$  labeled by  $\beta_p$ ; (b) introduce two vertices of  $F_{u_i}$  labeled by  $\gamma_p$ ; (c) introduce two vertices of  $F_{v_j}$  labeled  $\zeta_p$ ; (d) join the vertex labeled by  $\beta_p$  to the vertices labeled  $\gamma_p, \zeta_p$ .

Notice that we omitted the union operations from these descriptions. For each union operation in the expression tree for  $I(G)$ , we do as follows. If both graphs contain vertices labeled  $\alpha$ , then (a) vertices labeled  $\alpha_p$  in one of the graphs are relabeled  $\alpha'_p$  for  $p \in \{1, \dots, w\}$ ; (b) we perform the union operation; (c) the vertices labeled  $\alpha_p$  and  $\alpha'_q$  are joined for  $p, q \in \{1, \dots, w\}$ ; and (d) the vertices labeled  $\alpha'_p$  are relabeled  $\alpha_p$  for  $p \in \{1, \dots, w\}$ .

If both graphs contain vertices of the same gadget  $F_{u_i}$  labeled  $\gamma$ , then we again modify the tree by replacing the union operation by recursively repeating the following for  $i = 1, \dots, n$ : if both graphs contain vertices of  $F_{u_i}$  labeled  $\gamma$ , then (a) vertices labeled  $\gamma_p$  in one of the graphs are relabeled  $\gamma'_p$  for  $p \in \{1, \dots, w\}$  if  $\gamma_p$  is a label of a vertex of  $F_{u_i}$ ; (b) we perform the union operation; (c) the vertices labeled  $\gamma_p$  and  $\gamma'_q$  are joined for  $p, q \in \{1, \dots, w\}$  if  $\gamma_p, \gamma'_q$  are labels of vertices of  $F_{u_i}$ ; and (d) the vertices labeled  $\gamma'_p$  are relabeled  $\gamma_p$  for  $p \in \{1, \dots, w\}$ .

Similarly, if both graphs contain vertices of the same gadget  $F_{v_i}$  labeled  $\zeta$ , then we proceed by replacing the union operation by recursively repeating the following for  $i = 1, \dots, r$ : if both graphs contain vertices of  $F_{v_i}$  labeled  $\zeta$ , then (a) vertices labeled  $\zeta_p$  in one of the graphs are relabeled  $\zeta'_p$  for  $p \in \{1, \dots, w\}$  if  $\zeta_p$  is a label of a vertex of  $F_{v_i}$ ; (b) we perform the union operation; (c) the vertices labeled  $\zeta_p$  and  $\zeta'_q$  are joined for  $p, q \in \{1, \dots, w\}$  if  $\zeta_p, \zeta'_q$  are labels of vertices of  $F_{v_i}$ ; and (d) the vertices labeled  $\zeta'_p$  are relabeled  $\zeta_p$  for  $p \in \{1, \dots, w\}$ .

In all other cases we just do the union operation.

If, in the expression tree of  $I(G)$ , we have a join operation between two labels, say  $\mu_p$  and  $\mu_q$ , then we do the following. The vertices labeled  $\alpha_p$  and  $\beta_q$  are joined and symmetrically we join the vertices labeled  $\alpha_q$  and  $\beta_p$ . For  $i = 1, \dots, n$ , if there are vertices of  $F_{u_i}$  labeled  $\gamma_p, \delta_q$  ( $\gamma_q, \delta_p$ , respectively) then (a) the vertices labeled  $\gamma_p, \delta_q$  ( $\gamma_q, \delta_p$ , respectively) are joined; (b) the vertices labeled  $\gamma_p$  ( $\gamma_q$ , respectively) are relabeled by  $\delta_p$  ( $\delta_q$ , respectively). For  $i = 1, \dots, r$ , if there are vertices of  $F_{v_i}$  labeled  $\zeta_p, \eta_q$  ( $\zeta_q, \eta_p$ , respectively) then (a) the vertices labeled  $\zeta_p, \eta_q$  ( $\zeta_q, \eta_p$ , respectively) are joined; (b) the vertices labeled  $\zeta_p$  ( $\zeta_q$ , respectively) are relabeled by  $\eta_p$  ( $\eta_q$ , respectively).

The relabel operation in the expression tree of  $I(G)$ , that is, relabel  $\mu_p$  to  $\mu_q$  is replaced by the following relabelings: (a)  $\alpha_p$  to  $\alpha_q$ ; (b)  $\beta_p$  to  $\beta_q$ ; (c)  $\gamma_p$  to  $\gamma_q$ ; (d)  $\delta_p$  to  $\delta_q$ ; (e)  $\zeta_p$  to  $\zeta_q$ ; (f)  $\eta_p$  to  $\eta_q$ .

After we have completed the scanning of the expression tree for  $I(G)$ , we have to complete the construction of  $F_R$ . It remains to create the vertices  $x_{i1}^R, x_{i2}^R$  for  $n < i \leq k + m + 1$  and join them to other sets of the  $(k + m + 1)$ -partition. We repeat  $k + m + 1 - n$  times the following: (a) introduce two vertices labeled  $\lambda$ ;

(b) join the vertices labeled  $\lambda$  to the vertices labeled  $\alpha_1, \dots, \alpha_w$ ; (c) relabel the vertices labeled  $\lambda$  by  $\alpha_1$ .  $\square$

## D. Algorithmic upper bounds for $d$ -Regular Induced Subgraph

We sketch a dynamic programming algorithm for solving  $d$ -REGULAR INDUCED SUBGRAPH in time  $d^{O(w)} \cdot n$  on graphs of clique-width at most  $w$ .

First, we describe what we store in the tables corresponding to the nodes in an expression tree. Let  $G$  be a graph with  $n$  vertices and let  $T$  be an expression tree for  $G$  of width  $w$ . Recall that for a node  $X$  of  $T$ , we denote by  $G_X$  the  $w$ -graph associated with this node. We let  $U_1(X), \dots, U_w(X)$  be the sets of vertices of  $G_X$  labeled  $1, \dots, w$  respectively. The table of data for the node  $X$  stores pairs of integer vectors,  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$ , satisfying  $0 \leq s_i \leq d + 1$  and  $0 \leq d_i \leq d$  for all  $i \in \{1, \dots, w\}$ , and for which there is an induced subgraph  $H$  of  $G_X$  such that for all  $i \in \{1, \dots, w\}$

- $d_i = d_H(v)$  for all  $v \in V(H) \cap U_i(X)$  (if  $V(H) \cap U_i(X) = \emptyset$  then it is assumed that this condition holds for all  $0 \leq d_i \leq d$ ), and
- $s_i = \min\{|V(H) \cap U_i(X)|, d + 1\}$ .

If  $X$  is the root node of  $T$  (that is,  $G = G_X$ ) then  $G$  contains a  $d$ -regular induced subgraph if and only if the table for  $X$  contains an entry with the vector  $(d_1, \dots, d_w) = (d, \dots, d)$ .

Now we give the details of how we construct our tables and how we update them.

**Introduce Node:** Tables for introduce nodes of  $T$  are constructed in a straightforward manner. Suppose that  $X = i(v)$  for  $v \in V(G)$  and  $i \in \{1, \dots, w\}$ . Then the table of data for the node  $X$  contains the pairs of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  such that  $s_j = 0$  and  $0 \leq d_j \leq d$  for  $j \in \{1, \dots, w\}$ ,  $j \neq i$ , and either  $s_i = 0$  and  $0 \leq d_i \leq d$  or  $s_i = 1$  and  $d_i = 0$ .

**Relabel Node:** Suppose that  $X$  is a relabel node  $\rho_{i \rightarrow j}$ , and let  $Y$  be the child of  $X$ . Then the table for  $X$  contains a pair of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  if and only if  $s_i = 0$ , and the table for  $Y$  contains the entry  $(s'_1, \dots, s'_w), (d'_1, \dots, d'_w)$  such that

- $s_p = s'_p$  and  $d_p = d'_p$  for  $p \in \{1, \dots, w\}$ ,  $p \neq i, j$ ,
- $d_j = d'_i = d'_j$ ,
- $s_j = \min\{s'_i + s'_j, d + 1\}$ .

**Union Node:** Let  $X$  be a union node with children  $Y$  and  $Z$ . In this case the table for  $X$  contains a pair of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  if and only if the tables for  $Y$  and  $Z$  have the pairs of vectors  $(s'_1, \dots, s'_w), (d_1, \dots, d_w)$  and  $(s''_1, \dots, s''_w), (d_1, \dots, d_w)$ , respectively, such that  $s_i = \min\{s'_i + s''_i, d + 1\}$  for  $i \in \{1, \dots, w\}$ .

**Join Node:** Finally, suppose that  $X$  is a join node  $\eta_{i,j}$  with the child  $Y$ . The table for  $X$  has a pair of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  if and only if the table for  $Y$  includes a pair of vectors  $(s_1, \dots, s_w), (d'_1, \dots, d'_w)$  such that

- $d'_p = d_p$  for  $p \in \{1, \dots, w\}, p \neq i, j$ ,
- $d_i = d'_i + s_j$  and  $d_j = d'_j + s_i$ .

Correctness of the algorithm follows from the description, and the following easy observation.

**Observation 1.** *Let  $H$  be an induced  $d$ -regular subgraph of  $G$ . Then for any node  $X$  of the expression tree  $T$ , all vertices of  $H[V(G_X)]$  from the set  $U_i(X)$  have the same degree (in  $H$ ) for  $i \in \{1, \dots, w\}$ .*

Since for each  $X$ , the table for  $X$  contains at most  $(d+2)^{2w}$  pairs of vectors, the algorithm runs in time  $d^{O(w)} \cdot n$ . This proves that  $d$ -REGULAR INDUCED SUBGRAPH can be solved in time  $d^{O(w)} \cdot n$  on graphs of clique-width at most  $w$ .

## E. Proof of Lemma 3

We need an additional technical lemma about properties of the gadget  $F(x)$ . They are summarized as follows.

**Lemma 6.**

- Let  $G'$  be the graph obtained from a graph  $G$  with a vertex  $x$  by adding a copy of  $F(x)$ . For any  $d$ -regular induced subgraph  $H$  of  $G'$ , if any vertex of  $F(x) - x$  is included in  $H$  then  $V(F(x)) \subseteq V(H)$  and if  $x \notin V(H)$  then  $V(F(x)) \cap V(H) = \emptyset$ .
- We have  $\mathbf{cwd}(F(x)) \leq 4$ , and a 4-graph isomorphic to  $F(x)$  can be constructed in such a way that one label is used only for the vertex  $x$ . Moreover it can be assumed that the construction starts from the vertex  $x$  with a given label  $\alpha$ , and at the end all non-root vertices are relabeled by a given label  $\lambda$ .

*Proof.* The first claim of the lemma follows immediately from the fact that all vertices of  $F(x)$  except  $x$  have degree  $d$ .

To prove the second claim, let us note that  $F(x)$  is a cograph (i.e. a graph without induced paths on four vertices). It is well-known that cographs have clique-width 2 (see e.g. [35]). It means that in fact  $\mathbf{cwd}(F(x)) = 2$ . Clearly, by using two additional labels  $\alpha$  and  $\lambda$ , we can get the desired 4-graph.  $\square$

Now we can prove Lemma 3.

**Lemma 3.** *The red-blue graph  $G$  has an exact saturated capacitated dominating set of size  $k$  if and only if  $H$  contains an induced  $d$ -regular subgraph.*

*Proof.* Let  $S$  be an exact saturated capacitated dominating set of size  $k$  in  $G$  and let  $f$  be a corresponding domination mapping. We construct an induced  $d$ -regular subgraph  $Q$  of  $H$  as follows:

- All the vertices  $v_1, \dots, v_r, a_1, \dots, a_s, x, y, b_1, \dots, b_t$  and  $z$  are included in the set of vertices of  $Q$ .
- For each  $u_i \in S$ , all the vertices  $u_i, c_{i1}, \dots, c_{il_i}$  and  $p_i$  are included in the set of vertices of  $Q$ .
- For each  $i \in \{1, \dots, r\}$ , the vertex  $w_e$  for  $e = v_i f(v_i) \in E(G)$  is included in the set of vertices of  $Q$ .
- For each gadget  $F(w)$  rooted at a vertex  $w \in V(H)$  such that  $w \in V(Q)$ , all the vertices of  $F(w)$  are included in  $Q$ .

It is straightforward to check that the subgraph of  $H$  induced by the vertices of  $Q$  is a  $d$ -regular graph.

Assume now that  $Q$  is a  $d$ -regular induced subgraph of  $H$ . First we need some properties of  $Q$ .

**Claim 3.** *All the vertices  $v_1, \dots, v_r, a_1, \dots, a_s, x, y, b_1, \dots, b_t, z \in V(Q)$ .*

*Proof (of Claim 3).* Assume for the sake of contradiction that at least one vertex of the set  $\{v_1, \dots, v_r, a_1, \dots, a_s, x\}$  is not in  $V(Q)$ . Then all the vertices  $a_1, \dots, a_s, x, y$  are not in  $V(Q)$  since they have degree  $d$  in  $H$ . Each vertex  $v_i$  is adjacent to at most  $n$  vertices  $w_e$  and hence  $d_H(v_i) \leq n + r + s \leq d - 1 + s$ . Hence if  $a_1, \dots, a_s \notin V(Q)$  then  $v_1, \dots, v_r \notin V(Q)$ . Now we can conclude that all the vertices  $w_e$  cannot be vertices of  $Q$  and therefore  $u_1, \dots, u_n \notin V(Q)$ . Since  $y \notin V(Q)$ , by the same arguments,  $b_1, \dots, b_t, z \notin V(Q)$ , and if  $u_1, \dots, u_n \notin V(Q)$  then all vertices  $p_i$  and  $c_{ij}$  also cannot be included in  $V(Q)$ . But this means that  $V(Q) = \emptyset$ . We easily get the same conclusion if we assume that at least one vertex of the set  $\{y, z, b_1, \dots, b_t\}$  is not in  $V(Q)$ .  $\square$

**Claim 4.** *There is a  $k$ -element set  $I \subseteq \{1, \dots, n\}$  such that for any  $i \in I$ , we have  $u_i, c_{i1}, \dots, c_{il_i}, p_i \in V(Q)$ , and for any  $i \notin I$ , we have  $u_i, c_{i1}, \dots, c_{il_i}, p_i \notin V(Q)$ .*

*Proof (of Claim 4).* By Claim 3, we have  $y, b_1, \dots, b_t, z \in V(Q)$ . Hence  $z$  should be adjacent in  $Q$  to at least  $k$  other vertices, and therefore we have that exactly  $k$  vertices  $p_{i_1}, \dots, p_{i_k} \in V(Q)$ . Let  $I = \{i_1, \dots, i_k\}$ . It remains to observe that if  $p_j \in V(Q)$  then  $p_j, c_{j1}, \dots, c_{jl_j}, u_j \in V(Q)$ , and  $p_j, c_{j1}, \dots, c_{jl_j}, u_j \notin V(Q)$  otherwise.  $\square$

By Claim 3, we have  $v_1, \dots, v_r, a_1, \dots, a_s, x \in V(Q)$ . It follows that each vertex  $v_i$  is adjacent to exactly one vertex  $w_e$  in  $Q$  for some edge  $e = v_i u_j \in E(G)$ , and  $u_j \in V(Q)$  since  $d_H(w_e) = d$ . By Claim 4, we have that exactly  $k$  vertices  $u_{i_1}, \dots, u_{i_k} \in V(Q)$ . Also by this claim,  $u_{i_j}, c_{i_j 1}, \dots, c_{i_j l_{i_j}}, p_{i_j} \in V(Q)$  and hence each vertex  $u_{i_j}$  should be adjacent to exactly  $c(u_{i_j})$  vertices  $w_e$ . We let  $S = \{u_{i_1}, \dots, u_{i_k}\} \subseteq R$  in the red-blue graph  $G$  and define a mapping  $f: B \rightarrow R$  by setting  $f(v_i) = u_j$ . It remains to observe that  $S$  is an exact saturated capacitated dominating set and  $f$  is a domination mapping for this set.  $\square$

## F. Proof of Lemma 4

Now we show that the clique-width of  $H$  is bounded from above by a linear function in the clique-width of the incidence graph  $I(G)$  of  $G$ .

**Lemma 4.** *We have that  $\mathbf{cwd}(H) \leq 3 \cdot \mathbf{cwd}(I(G)) + 6$  and an expression tree of width at most  $3 \cdot \mathbf{cwd}(I(G)) + 6$  for  $H$  can be constructed in polynomial time assuming we are given an expression tree of width  $\mathbf{cwd}(I(G))$  for  $I(G)$ .*

*Proof.* Let  $w = \mathbf{cwd}(I(G))$  and suppose that the expression tree for  $I(G)$  uses labels  $\{\alpha_1, \dots, \alpha_w\}$ . To construct an expression tree for  $H$  we use the following additional labels:

- labels  $\beta_1, \dots, \beta_w$  and  $\gamma_1, \dots, \gamma_w$  for the vertices  $v_1, \dots, v_r$ ;
- a label  $\delta$  for vertices  $p_i$ ;
- working labels  $\zeta_1, \zeta_2$  for the vertices  $a_1, \dots, a_s, b_1, \dots, b_t, c_{i1}, \dots, c_{il_i}, x, y$  and  $z$ ;
- working labels  $\eta_1, \eta_2$  for non-root vertices of gadgets  $F(w)$ ; and
- a label  $\lambda$  for vertices of  $F(w)$  and for vertices that are not joined to other vertices in further stages of the construction.

We construct the required expression tree for  $H$  by going over the expression tree for  $I(G)$  and making necessary changes to it.

When a vertex  $u_i \in R$  labeled by  $\alpha_p$  is introduced, we perform the following set of operations. We first introduce the vertex  $u_i$  labeled  $\alpha_p$ . Then  $l_i$  vertices  $c_{i1}, \dots, c_{il_i}$  are created by repeating the following operations  $l_i$  times: (a) introduce a vertex labeled  $\zeta_1$ ; (b) join vertices labeled  $\zeta_1$  to the vertices labeled  $\alpha_p$  and  $\zeta_2$ ; (c) construct copies of  $F(c_{ij})$  rooted at this vertex using the labels  $\eta_1, \eta_2$  as it is shown in Lemma 6 (recall that when a copy of  $F(u_i)$  is constructed here and further, all the non-root vertices are relabeled by  $\lambda$ ); (d) relabel vertices labeled  $\zeta_1$  by  $\zeta_2$ . Finally, (a) the vertex  $p_i$  labeled by  $\delta_1$  is introduced; (b) the vertex is joined to the vertices labeled  $\alpha_p$  and  $\zeta_2$ ; (c) copies of  $F(p_i)$  rooted at this vertex are constructed; (d) all the vertices labeled  $\zeta_2$  are relabeled  $\lambda$ . 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 of  $I(G)$  that corresponds to an edge  $e \in E(G)$  labeled  $\alpha_p$  is introduced, we introduce the vertex  $w_e$  also labeled  $\alpha_p$  and add copies of  $F(w_e)$  rooted at this vertex.

When a vertex  $v_i \in B$  labeled by  $\alpha_p$  is introduced, we introduce the vertex  $v_i$  labeled  $\beta_p$ .

For each union operation in the expression tree for  $I(G)$ , we do as follows. If both graphs contain vertices labeled  $\beta$ , then (a) vertices labeled  $\beta_p$  in one of the graphs are relabeled  $\gamma_p$  for  $p \in \{1, \dots, w\}$ ; (b) we perform the union operation; (c) the vertices labeled  $\beta_p$  and  $\gamma_q$  are joined for  $p, q \in \{1, \dots, w\}$ ; and (d) the vertices labeled  $\gamma_p$  are relabeled  $\beta_p$  for  $p \in \{1, \dots, w\}$ . If only one graph contains vertices labeled  $\beta$  then we just do the union operation.

If, in the expression tree of  $I(G)$ , we have a join operation between two labels, say  $\alpha_p$  and  $\alpha_q$ , then we simulate this by applying the join operations between the following: (a)  $\alpha_p$  and  $\alpha_q$ ; (b)  $\alpha_p$  and  $\beta_q$ ; (c)  $\beta_p$  and  $\alpha_q$ .

A relabel operation in the expression tree of  $I(G)$ , say an operation relabeling  $\alpha_p$  to  $\alpha_q$ , is replaced by the following relabeling process: (a)  $\alpha_p$  to  $\alpha_q$  and (b)  $\beta_p$  to  $\beta_q$ .

After we have completed the scanning of the expression tree for  $I(G)$ , we complete the construction of  $H$ . We create vertices  $a_1, \dots, a_s$  by repeating the following operations  $s$  times: (a) introduce a vertex labeled  $\zeta_1$ ; (b) join vertices labeled  $\zeta_1$  to the vertices labeled by labels  $\beta_p$  for  $p \in \{1, \dots, w\}$  and  $\zeta_2$ ; (c) construct copies of  $F(a_i)$  rooted at this vertex using the labels  $\eta_1, \eta_2$ ; (d) relabel vertices labeled  $\zeta_1$  by  $\zeta_2$ . Then (a) the vertex  $x$  labeled by  $\zeta_1$  is introduced; (b) it is joined to all the vertices labeled  $\zeta_2$  and  $\beta_p$  for  $p \in \{1, \dots, w\}$ ; (c) all the vertices labeled  $\zeta_2$  are relabeled by  $\lambda$ . Now (a) the vertex  $y$  labeled by  $\zeta_2$  is introduced; (b) copies of  $F(y)$  rooted at this vertex are added; (c)  $y$  is joined to the vertex  $x$  labeled  $\zeta_1$ ; (c) all the vertices labeled  $\zeta_1$  are relabeled by  $\lambda$ . Then we create vertices  $b_1, \dots, b_t$  by repeating the following operations  $t$  times: (a) introduce a vertex labeled  $\zeta_1$ ; (b) join vertices labeled  $\zeta_1$  to vertices labeled  $\zeta_2$ ; (c) construct copies of  $F(b_i)$  rooted at this vertex; (d) relabel vertices labeled  $\zeta_1$  by  $\zeta_2$ . Finally we (a) introduce the vertex  $z$  labeled  $\zeta_1$  and (b) join it to all the vertices labeled by  $\zeta_2$  or  $\delta_1$ .  $\square$

## G. Proof of Corollary 3

**Corollary 3.** *The MINIMUM  $d$ -REGULAR INDUCED SUBGRAPH, MAXIMUM  $d$ -REGULAR INDUCED SUBGRAPH and COUNTING  $d$ -REGULAR INDUCED SUBGRAPH problems can be solved on  $n$ -vertex graphs of clique-width at most  $w$  in time  $d^{O(w)} \cdot n$  if an expression tree of width  $w$  is given, but they cannot be solved in time  $2^{o(w \log n)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

*Proof.* To prove the algorithmic upper bounds, it is sufficient to modify the algorithm described for  $d$ -REGULAR INDUCED SUBGRAPH. Particularly for MINIMUM  $d$ -REGULAR INDUCED SUBGRAPH, each entry of the table of data for the node  $X$  should store a positive integer  $p$  and pairs of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  of integers such that  $0 \leq s_i \leq d+1$  and  $0 \leq d_i \leq d$  for  $i \in \{1, \dots, w\}$ , for which  $p$  is the number of vertices of an induced subgraph  $H$  of  $G_X$  of minimum size such that for  $i \in \{1, \dots, w\}$

- $d_i = d_H(v)$  for all  $v \in V(H) \cap U_i(X)$  (observe that if  $V(H) \cap U_i(X) = \emptyset$  then it is assumed that this condition holds for any  $0 \leq d_i \leq d$ ), and
- $s_i = \min\{|V(H) \cap U_i(X)|, d+1\}$ .

For MAXIMUM  $d$ -REGULAR INDUCED SUBGRAPH and COUNTING  $d$ -REGULAR INDUCED SUBGRAPH, the parameter  $p$  should be the number of vertices of an induced subgraph of maximum size and the number of subgraphs, respectively.

The lower bounds follow from Theorem 2 immediately.  $\square$

## H. Proof of Theorem 3

**Theorem 3.** *The MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem can be solved on  $n$ -vertex graphs of clique-width at most  $w$  in time  $d^{O(w)} \cdot n$  if an expression tree of width  $w$  is given, but it cannot be solved in time  $2^{o(w \log d)} \cdot n^{O(1)}$  unless ETH fails, even if an expression tree of width  $w$  is given.*

### H.1 Algorithmic upper bounds for Minimum Subgraph of Minimum Degree at least $d$

The MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem can be solved by a variant of the dynamic programming algorithm for  $d$ -REGULAR INDUCED SUBGRAPH.

First we describe what we store in the tables corresponding to the nodes in the expression tree. Let  $G$  be a graph with  $n$  vertices and let  $T$  be an expression tree for  $G$  of width  $w$ . Let  $U_1(X), \dots, U_w(X)$  be the sets of vertices of  $G_X$  labeled  $1, \dots, w$ , respectively. Each entry of the table of data for the node  $X$  stores a positive integer  $p$  and pairs of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  of integers such that  $0 \leq s_i \leq d$  and  $0 \leq d_i \leq d$  for  $i \in \{1, \dots, w\}$ , for which  $p$  is the number of vertices of an induced subgraph  $H$  of  $G_X$  of minimum size such that for  $i \in \{1, \dots, w\}$

- $d_i = \min\{d, \min\{d_H(v) \mid v \in V(H) \cap U_i(X)\}\}$  (it is assumed that  $d_i = d$  if  $V(H) \cap U_i(X) = \emptyset$ ), and
- $s_i = \min\{|V(H) \cap U_i(X)|, d\}$ .

If  $X$  is the root node of  $T$  (that is,  $G = G_X$ ) then  $G$  contains an induced subgraph of degree at least  $d$  with at most  $k$  vertices if and only if the table for  $X$  contains an entry with the parameter  $p \leq k$  and vector  $(d_1, \dots, d_w)$  such that  $d_i \geq d$  for  $i \in \{1, \dots, w\}$ .

Now we give the details of how we construct our tables and how we update them.

**Introduce Node:** Tables for introduce nodes of  $T$  are constructed in a straightforward manner. Suppose that  $X = i(v)$  for  $v \in V(G)$  and  $i \in \{1, \dots, w\}$ . Then the table of data for the node  $X$  contains the entries for  $p = 0$  and  $p = 1$ . For  $p = 0$ , the table stores the pairs of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  such that  $s_j = d$  and  $d_i = d$  for  $j \in \{1, \dots, w\}$ . For  $p = 1$ , the table contains the pairs of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  such that  $s_j = d$  and  $d_i = d$  for  $j \in \{1, \dots, w\}$ ,  $j \neq i$ , and  $s_i = 1$ ,  $d_i = 0$ .

**Relabel Node:** Suppose that  $X$  is a relabel node  $\rho_{i \rightarrow j}$ , and let  $Y$  be the child of  $X$ . Then the table for  $X$  contains an entry with an integer  $p$  and a pair of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  if and only if  $s_i = 0$  and  $p$  is the minimum integer for which the table for  $Y$  contains the entry with  $p$  and the vectors  $(s'_1, \dots, s'_w)$ ,  $(d'_1, \dots, d'_w)$  such that

- $s'_q = s_q$  and  $d'_q = d_q$  for  $q \in \{1, \dots, w\}$ ,  $q \neq i, j$ ,

- $d_i = \min\{d'_i, d'_j\}$ ,
- $s_j = \min\{s'_i + s'_j, d\}$ .

**Union Node:** Let  $X$  be a union node with children  $Y$  and  $Z$ . In this case the table for  $X$  contains an entry with  $p$  and a pair of vectors  $(s_1, \dots, s_w)$  and  $(d_1, \dots, d_w)$  if and only if  $p$  is the minimum integer for which the tables for  $Y$  and  $Z$  have the entries  $p'$ ,  $(s'_1, \dots, s'_w)$ ,  $(d'_1, \dots, d'_w)$  and  $p''$ ,  $(s''_1, \dots, s''_w)$ ,  $(d''_1, \dots, d''_w)$ , respectively, such that

- $p = p' + p''$ ,
- $d_i = \min\{d'_i, d''_i\}$  for  $i \in \{1, \dots, w\}$ , and
- $s_i = \min\{s'_i + s''_i, d\}$  for  $i \in \{1, \dots, w\}$ .

**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 an entry  $p$ ,  $(s_1, \dots, s_w)$ ,  $(d_1, \dots, d_w)$  if and only if  $p$  is the minimum integer for which the table for  $Y$  includes the entry  $p$ ,  $(s_1, \dots, s_w)$ ,  $(d'_1, \dots, d'_w)$  such that

- $d'_q = d_q$  for  $q \in \{1, \dots, w\}$ ,  $q \neq i, j$ ,
- $d_i = \min\{d'_i + s_j, d\}$  and  $d_j = \min\{d'_j + s_i, d\}$ .

Correctness of the algorithm follows from the description of the procedure.

Since for each  $X$ , the table for  $X$  contains at most  $(d+1)^{2w}$  pairs of vectors and for each pair of vectors only one value of the parameter  $p$  is stored, the algorithm runs in time  $d^{O(w)} \cdot n$ . This proves that MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  can be solved in time  $d^{O(w)} \cdot n$  on graphs of clique-width at most  $w$ .

## H.2 Lower bounds

To prove our lower bounds we give a reduction from the RED-BLUE CDS problem, parameterized by the clique-width of the incidence graph of the input graph, to the MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  problem. The proof is based on the same ideas as the proof for the  $d$ -REGULAR INDUCED SUBGRAPH problem.

Again we first give a construction, then prove its correctness, and finally prove that the clique-width of the graph in the reduced instance is bounded by a linear function in the clique-width of the incidence graph of the original graph.

**Construction.** Let  $(G, c, k)$  be an instance of RED-BLUE CDS with  $R = \{u_1, \dots, u_n\}$  being the set of red vertices and  $B = \{v_1, \dots, v_r\}$  being the set of blue vertices. Let  $d = n + r + 2$  if  $n + r$  is odd and let  $d = n + r + 3$  otherwise; notice that  $d$  is odd.

We need some auxiliary gadgets. Here we use the gadget  $F(x)$  defined in the proof of Theorem 2. Observe that  $F(x)$  has  $f = \frac{(d+1)(d-1)}{2} + 1$  non-root vertices of  $F(x)$ . Now we construct graphs  $Q_i(u, x, D)$  for  $i \in \{1, \dots, n\}$ .

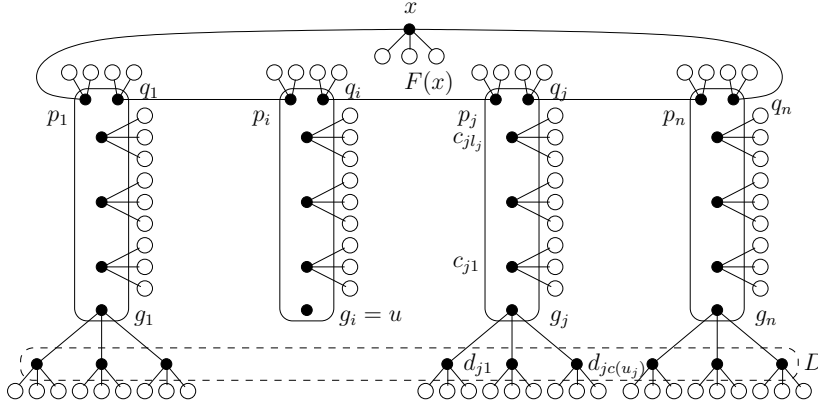
**Auxiliary gadget  $Q_i(u, x, D)$ :**

1. For each  $j \in \{1, \dots, n\}$ , let  $l_j = d - c(u_j) - 2$  and do the following:
  - Introduce two adjacent vertices  $p_j, q_j$ , and construct  $c(u_j) - 1$  copies of  $F(p_j)$  and  $F(q_j)$ .



- Introduce a vertex  $g_j$  and join it to  $p_j$  and  $q_j$ .
  - Construct a clique of size  $l_j$  with vertices  $c_{j1}, \dots, c_{jl_j}$ , join them to the vertices  $g_j, p_j$  and  $q_j$  by edges, and for each  $h \in \{1, \dots, l_j\}$ , introduce  $c(u_j)$  copies of  $F(c_{jh})$ .
2. For each  $j \in \{1, \dots, n\}, j \neq i$ , do the following:
    - Introduce vertices  $d_{j1}, \dots, d_{jc(u_j)}$  and join them to  $g_j$ .
    - Construct  $d - 2$  copies of  $F(d_{jh})$  for  $h \in \{1, \dots, c(u_j)\}$ .
  3. For  $j \in \{1, \dots, n - 1\}$ , the edges  $q_j p_{j+1}$  are introduced.
  4. A vertex  $x$  is introduced,  $d - 3$  copies of  $F(x)$  are constructed, and finally  $x$  is joined by edges to  $p_1$  and  $q_n$ .

Now let  $u = g_i$  and let  $D = \{d_{jh} | 1 \leq j \leq n, j \neq i, 1 \leq h \leq l_j\}$ . We attach gadgets  $Q_i(u, x, D)$  to other parts of our construction by the vertices  $u, x$  and join  $D$  to other vertices in such a way that all the vertices of  $D$  are adjacent to one vertex. The construction of  $Q_i(u, x, D)$  is illustrated in Figure 3. The properties of  $Q_i(u, x, D)$  are given in the following lemma.



**Fig. 3.** Construction of  $Q_i(u, x, D)$ .

**Lemma 7.**

- Let  $G'$  be the graph obtained from a graph  $G$  with vertices  $u, x$  and  $w$  by adding a copy of  $Q_i(u, x, D)$  and joining  $w$  to all the vertices of  $D$  by edges. For any induced subgraph  $H$  of  $G'$  of minimum degree at least  $d$ , if any vertex of  $Q_i(u, x, D) - \{u, x\}$  is included in  $H$  then  $V(Q_i(u, x, D)) \subseteq V(H)$  and if  $u \notin V(H)$  or  $x \notin V(H)$  then  $(V(Q_i(u, x, D)) \setminus \{u, x\}) \cap V(H) = \emptyset$ .
- We have that  $\mathbf{cwd}(Q_i(u, x, D)) \leq 10$ , and the 10-graph isomorphic with  $Q_i(u, x, D)$  can be constructed in such a way that one given label  $\alpha$  is used only for the vertex  $u$ , another given label  $\beta$  is used only for  $x$ , all the vertices of  $D$  are relabeled by a given label  $\gamma$ , and all other vertices are relabeled by a given label  $\lambda$  at the end of the construction.

$$- |V(Q_i(u, x, D))| = q - c(u_i)(d - 2)f \text{ where } q = \sum_{j=1}^n [(2dc(u_j) - c(u_j)^2 - 2c(u_j) - 2)f + 1].$$

*Proof.* The first claim of the lemma follows immediately from the fact that all the non-root vertices of  $Q_i(u, x, D)$  have degree  $d$ .

To prove the second claim, we sketch how to construct a 10-graph isomorphic to  $Q_i(u, x, D)$  using the following labels:

- labels  $\alpha, \beta$  for the vertices  $u$  and  $x$ , respectively;
- labels  $\gamma$  and  $\eta$  for the vertices  $d_{jh}$ ;
- labels  $\delta_1, \delta_2$  for the vertices  $p_j, q_j, c_{jh}$  and  $q_j$  ( $j \neq i$ );
- labels  $\zeta_1, \zeta_2$  for the non-root vertices of the gadgets  $F(w)$ ; and
- label  $\lambda$  for vertices that are not joined to other vertices in the further stages of the construction.

First, we introduce the vertex  $x$  labeled  $\alpha$ .

For  $j = 1, \dots, i-1$  we do the following. We introduce the vertex  $p_j$  labeled by  $\delta_1$ . If  $j > 1$  then we join it to the vertex  $q_{j-1}$  labeled  $\delta_2$  and relabel the vertices labeled  $\delta_2$  by  $\lambda$ . If  $j = 1$  then the vertex  $x$  labeled  $\alpha$  is joined to the vertex  $p_j$ . Then  $c(u_j) - 1$  copies of  $F(p_j)$  rooted at  $p_j$  are created using the labels  $\zeta_1, \zeta_2$  as described in Lemma 6 (recall that when a copy of  $F(w)$  is constructed, all the non-root vertices are relabeled by  $\lambda$ ). After that the vertex  $p_j$  is relabeled by  $\delta_2$ . Next, we repeat the following operations  $l_j$  times. (a) introduce the vertex  $c_{jh}$  labeled  $\delta_1$ ; (b) join vertices labeled  $\delta_1$  to the vertices  $\delta_2$ ; (c) construct copies of  $F(c_{jh})$  rooted at this vertex using the labels  $\zeta_1, \zeta_2$ ; (c) relabel vertices labeled  $\delta_1$  by  $\delta_2$ . Next, the vertex  $g_j$  labeled by  $\delta_1$  is added and joined to vertices labeled  $\delta_2$ . Then we deal with the vertices  $d_{jh}$  by repeating the following  $c(u_j)$  times: (a) create a vertex labeled  $\eta$ ; (b) join it to  $g_j$  labeled  $\delta_1$ ; (c) create  $d - 2$  copies of  $F(d_{jh})$  rooted at this vertex; (d) relabel the vertices labeled by  $\eta$  by  $\gamma$ . Finally, we (a) relabel  $g_j$  by  $\delta_2$ ; (b) introduce  $q_j$  labeled  $\delta_1$ ; (c) join it to all the vertices labeled  $\delta_2$ ; (d) construct  $c(u_j) - 1$  copies of  $F(q_j)$ ; (e) relabel all the vertices labeled  $\delta_2$  by  $\lambda$ ; (f) relabel  $q_j$  by  $\delta_2$ .

For  $j = i$ , we use almost the same operations. The only differences are that we label  $u = g_i$  by  $\beta$ , we do not relabel it, we do not repeat the operations described above for  $d_{jh}$ , and finally we join  $q_i$  to the vertex labeled  $\beta$ . For  $j = i + 1, \dots, n$ , our construction is the same as for  $j \in \{1, \dots, i - 1\}$ . The construction is concluded by joining  $x$  labeled by  $\alpha$  to  $q_n$  labeled by  $\gamma_2$  and relabeling  $q_n$  by  $\lambda$ .

The third claim of the lemma follows immediately from the description of  $Q_i(u, x, D)$ .  $\square$

**Reduction:** Now we describe our reduction. Let  $s = d - r - 2$  and  $t = d - k - 1$ .

1. Vertices  $u_1, \dots, u_n, h_1, \dots, h_n$  are introduced and for each  $i \in \{1, \dots, n\}$  a copy of a gadget  $Q_i(u_i, h_i, D_i)$  is created ( $u_i, h_i$  are the roots, and by  $D_i$  we denote the set of vertices  $D$  for this copy of a gadget).
2. A clique of size  $r$  with vertices  $v_1, \dots, v_r$  is created.

3. For each edge  $e = u_i v_j$  of  $G$ , a vertex  $w_e$  is added, joined by edges to  $u_i$  and  $v_j$ , and  $d - 2$  copies of  $F(w_e)$  are constructed.
4. A clique of size  $s$  with vertices  $a_1, \dots, a_s$  is created, all vertices  $a_i$  are joined to vertices  $v_1, \dots, v_r$ , and for each  $i \in \{1, \dots, s\}$ , a copy of  $F(a_i)$  is added.
5. A vertex  $x$  is introduced and joined by edges to  $v_1, \dots, v_r$  and  $a_1, \dots, a_s$ .
6. A vertex  $w$  is introduced and joined to  $v_1, \dots, v_r, a_1, \dots, a_s, x$  and all the vertices of the sets  $D_1, \dots, D_n$ .
7. A vertex  $y$  is introduced and joined by an edge to  $x$ , and  $k - 1$  copies of  $F(y)$  are added.
8. A clique of size  $t$  with vertices  $b_1, \dots, b_t$  is created, the vertex  $y$  is joined by edges to all the vertices of the clique, and for each  $j \in \{1, \dots, t\}$ ,  $k$  copies of  $F(b_j)$  are added.
9. A vertex  $z$  is introduced and joined by edges to the vertices  $y, b_1, \dots, b_t$  and all the vertices  $h_1, \dots, h_n$ .

Denote the obtained graph by  $H$  (see Figure 4).

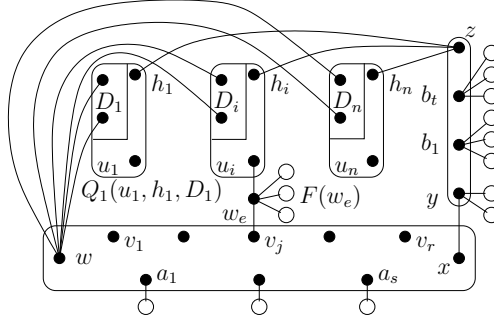


Fig. 4. Construction of  $H$ .

**Lemma 8.** *The red-blue graph  $G$  has a capacitated dominating set of size at most  $k$  if and only if  $H$  contains an induced subgraph with minimum degree at least  $d$  which has at most  $kq + n + s(f + 1) + (k - 1)f + t(kf + 1) + 4$  vertices.*

*Proof.* Suppose that the red-blue graph  $G$  has a capacitated dominating set of size at most  $k$ . Clearly, we can assume that there is a capacitated dominating set  $S$  of size exactly  $k$ . Let  $f$  be a corresponding domination mapping. We construct an induced subgraph  $J$  of  $H$  as follows:

- Include all the vertices  $v_1, \dots, v_r, a_1, \dots, a_s, x, w, y, b_1, \dots, b_t$  and  $z$  in  $J$ .
- For each  $u_i \in S$ , all the vertices of  $Q_i(u_i, h_i, D_i)$  are included in  $J$ .
- For each  $i \in \{1, \dots, r\}$ , the vertex  $w_e$  for  $e = v_i f(v_i) \in E(G)$  is included in the set of vertices of  $J$ . For each  $j \in \{1, \dots, n\}$ , if  $|f^{-1}(u_j)| < c(u_j)$  then  $c(u_j) - |f^{-1}(u_j)| < c(u_j)$  additional vertices  $w_e$  which correspond to the edges incident with  $u_j$  in  $G$  are arbitrarily chosen and included in  $J$ .

- For each gadget  $F(p)$  rooted at a vertex  $p \in V(J)$ , all the non-root vertices of  $F(p)$  are included in  $J$ .

It is straightforward to check that  $J$  has  $kq+n+s(f+1)+(k-1)f+t(kf+1)+4$  vertices and  $d_J(p) \geq d$  for  $p \in V(J)$ .

Assume now that  $J$  is an induced subgraph of  $H$  with at most  $kq+n+s(f+1)+(k-1)f+t(kf+1)+4$  vertices such that  $d_J(p) \geq d$  for  $p \in V(J)$ . We need some properties of  $J$ .

**Claim 5.** *All the vertices  $v_1, \dots, v_r, a_1, \dots, a_s, x, w, y, b_1, \dots, b_t, z \in V(J)$ .*

*Proof (of Claim 5).* Assume for the sake of contradiction that at least one vertex of the set  $\{v_1, \dots, v_r, a_1, \dots, a_s, x, w\}$  is not in  $V(J)$ . Then all the vertices  $a_1, \dots, a_s, x, y \notin V(J)$  since they have degree  $d$  in  $H$ . Each vertex  $v_i$  is adjacent to at most  $n$  vertices  $w_e$  and hence  $d_H(v_i) \leq n+r+s+1 \leq d-1+s$ . Hence if  $a_1, \dots, a_s \notin V(J)$  then  $v_1, \dots, v_r \notin V(J)$ . Now we see that all the vertices  $w_e$  cannot be vertices of  $J$  and therefore  $u_1, \dots, u_n \notin V(J)$ . By the first claim of Lemma 7, all the vertices of the gadgets  $Q(u_i, h_i, D_i)$  are not included in  $J$  and also  $w \notin V(J)$ . Since  $y \notin V(J)$ , by similar arguments  $b_1, \dots, b_t, z \notin V(J)$ . This means that  $V(J) = \emptyset$ . We easily get the same conclusion if we assume that at least one vertex of the set  $\{y, z, b_1, \dots, b_t\}$  is not in  $J$ .  $\square$

**Claim 6.** *There is a  $k$ -element set  $I \subseteq \{1, \dots, n\}$  such that for any  $i \in I$ ,  $V(Q_i(u_i, h_i, D_i)) \subseteq V(J)$ , and for any  $i \notin I$ , none of the vertices of  $Q_i(u_i, h_i, D_i)$  is included in  $J$ .*

*Proof (of Claim 6).* By Claim 5,  $y, b_1, \dots, b_t, z \in V(J)$ . Hence  $z$  is adjacent in  $J$  to at least  $k$  other vertices, and therefore there are  $k' \geq k$  vertices  $h_{i_1}, \dots, h_{i_{k'}}$  in  $V(J)$ . By the first claim of Lemma 7, all the vertices of  $Q_{i_j}(u_{i_j}, h_{i_j}, D_{i_j})$  for  $j \in \{1, \dots, k'\}$  are in  $J$ . Now we observe that for each  $j \in \{1, \dots, k'\}$ , at least  $c(u_{i_j})$  vertices  $w_e$  adjacent to  $u_{i_j}$  should be included in  $J$ . Using the third claim of Lemma 7, we conclude that  $J$  has at least  $k'q+n+s(f+1)+(k-1)f+t(kf+1)+4$  vertices. So,  $k' = k$  and we let  $I = \{i_1, \dots, i_{k'}\}$ .  $\square$

Now we complete the proof of the lemma. By Claim 5, we have that  $v_1, \dots, v_r, a_1, \dots, a_s, x \in V(J)$ . It follows that each vertex  $v_i$  is adjacent to at least one vertex  $w_e$  in  $J$  for some edge  $e = v_i u_j \in E(G)$ , and  $u_j \in V(J)$  since  $d_H(w_e) = d$ . We define a mapping  $f: B \rightarrow R$  by setting  $f(v_i) = u_j$ . By Claim 6, exactly  $k$  vertices  $u_{i_1}, \dots, u_{i_k} \in V(J)$ . We let  $S = \{u_{i_1}, \dots, u_{i_k}\} \subseteq R$  in the red-blue graph  $G$ . Also by this claim, all the vertices of  $Q_{i_1}(u_{i_1}, h_{i_1}, D_{i_1}), \dots, Q_{i_k}(u_{i_k}, h_{i_k}, D_{i_k})$  are included in  $V(J)$ , and hence each vertex  $u_{i_j}$  should be adjacent to at least  $c(u_{i_j})$  vertices  $w_e$ . Observe that if there is a vertex  $u_{i_j}$  which is adjacent to more than  $c(u_{i_j})$  vertices  $w_e$ , then  $|V(J)| > kq+n+s(f+1)+(k-1)f+t(kf+1)+4$ . Therefore each vertex  $u_{i_j}$  is adjacent to exactly  $c(u_{i_j})$  vertices  $w_e$ . This means that  $S$  is a capacitated dominating set and  $f$  is a domination mapping for this set.  $\square$

Our next aim is to give an upper bound for the clique-width of  $H$  as a linear function in the clique-width of the incidence graph  $I(G)$  of  $G$ .

**Lemma 9.** *We have  $\mathbf{cwd}(H) \leq 3 \cdot \mathbf{cwd}(I(G)) + 12$ , and an expression tree of width at most  $3 \cdot \mathbf{cwd}(I(G)) + 12$  for  $H$  can be constructed in polynomial time, given an expression tree of width  $\mathbf{cwd}(I(G))$  for  $I(G)$ .*

*Proof.* Let  $w = \mathbf{cwd}(I(G))$  and suppose that the expression tree for  $I(G)$  uses labels  $\{\alpha_1, \dots, \alpha_w\}$ . To construct an expression tree for  $H$  we use the following additional labels:

- labels  $\beta_1, \dots, \beta_w$  and  $\gamma_1, \dots, \gamma_w$  for the vertices  $v_1, \dots, v_r$ ;
- labels  $\delta_1, \delta_2$  for the vertices  $h_i$  and for the vertices in the sets  $D_i$ ;
- working labels  $\zeta_1, \zeta_2$  for the vertices  $a_1, \dots, a_s, b_1, \dots, b_t, c_{i1}, \dots, c_{it}, x, y, z$  and  $w$ ;
- working labels  $\eta_1, \dots, \eta_7$  for the non-root vertices of the gadgets  $F(w)$  and  $Q_i(u_i, h_i, D_i)$ ; and
- a label  $\lambda$  for vertices of  $F(w)$  and for vertices that are not joined to other vertices in further stages of the construction.

We construct the required expression tree for  $H$  by going over the expression tree for  $I(G)$  and making necessary changes to it.

When a vertex  $u_i \in R$  labeled by  $\alpha_p$  is introduced, we create  $Q_i(u_i, h_i, D_i)$  as described in Lemma 7 using the label  $\alpha_p$  for  $u_i$ , the label  $\delta_1$  for  $h_i$  and  $\delta_2$  for the vertices of  $D_i$ ; the labels  $\eta_1, \dots, \eta_7$  are used for the construction of  $Q_i(u_i, h_i, D_i)$  and  $\lambda$  is used for the final relabeling.

When a vertex of  $I(G)$  which corresponds to an edge  $e \in E(G)$  labeled  $\alpha_p$  is introduced, we introduce the vertex  $w_e$  (labeled  $\alpha_p$ ) and add  $d - 2$  copies of  $F(w_e)$  rooted at this vertex using labels  $\eta_1, \eta_2$  and  $\lambda$ .

When a vertex  $v_i \in B$  labeled by  $\alpha_p$  is introduced, we introduce the vertex  $v_i$  labeled  $\beta_p$ .

For each union operation in the expression tree for  $I(G)$ , we do the following. If both graphs contain vertices labeled  $\beta$ , then (a) vertices labeled  $\beta_p$  in one of the graphs are relabeled  $\gamma_p$  for  $p \in \{1, \dots, w\}$ ; (b) we perform the union operation; (c) the vertices labeled  $\beta_p$  and  $\gamma_q$  are joined for  $p, q \in \{1, \dots, w\}$ ; and (d) the vertices labeled  $\gamma_p$  are relabeled  $\beta_p$  for  $p \in \{1, \dots, w\}$ . If only one graph contains vertices labeled  $\beta$  then we just do the union operation.

If in the expression tree of  $I(G)$ , we have join operation between two labels, say  $\alpha_p$  and  $\alpha_q$ , then we simulate this by applying join operations between the following: (a)  $\alpha_p$  and  $\alpha_q$ ; (b)  $\alpha_p$  and  $\beta_q$ ; (c)  $\beta_p$  and  $\alpha_q$ .

A relabel operation in the expression tree of  $I(G)$ , say relabeling  $\alpha_p$  to  $\alpha_q$ , is replaced by the following relabeling process: (a)  $\alpha_p$  to  $\alpha_q$  and (b)  $\beta_p$  to  $\beta_q$ .

After we have completed the scanning of the expression tree for  $I(G)$ , we perform the following operations to complete the construction of  $H$ . We create vertices  $a_1, \dots, a_s$  by repeating the following operations  $s$  times: (a) introduce a vertex labeled  $\zeta_1$ ; (b) join vertices labeled  $\zeta_1$  to the vertices labeled  $\beta_p$  for  $p \in \{1, \dots, w\}$  and  $\zeta_2$ ; (c) construct copies of  $F(a_i)$  rooted at this vertex using the labels  $\eta_1, \eta_2$ ; (d) relabel vertices labeled  $\zeta_1$  by  $\zeta_2$ . Then (a) the vertex  $w$  labeled by  $\zeta_1$  is introduced; (b) it is joined to all the vertices labeled  $\zeta_2, \delta_2$  and

$\beta_p$  for  $p \in \{1, \dots, w\}$ ; (c) it is relabeled by  $\zeta_2$ . Now (a) the vertex  $x$  labeled by  $\zeta_1$  is introduced; (b) it is joined to all the vertices labeled  $\zeta_2$  and  $\beta_p$  for  $p \in \{1, \dots, w\}$ ; (c) all the vertices labeled  $\zeta_2$  are relabeled by  $\lambda$ .

Now (a) the vertex  $y$  labeled by  $\zeta_2$  is introduced; (b) the copies of  $F(y)$  rooted at this vertex are added; (c) the vertex is joined to the vertex  $x$  labeled  $\zeta_1$ ; (c) all the vertices labeled  $\zeta_1$  are relabeled by  $\lambda$ . Then we create vertices  $b_1, \dots, b_t$  by repeating the following operations  $t$  times: (a) introduce a vertex labeled  $\zeta_1$ ; (b) join vertices labeled  $\zeta_1$  to the vertices labeled  $\zeta_2$ ; (c) construct copies of  $F(b_i)$  rooted at this vertex; (d) relabel vertices labeled  $\zeta_1$  by  $\zeta_2$ . Finally we (a) introduce the vertex  $z$  labeled  $\zeta_1$  and (b) it is joined to all the vertices labeled by  $\zeta_2$  or  $\delta_1$ .  $\square$

To finish the proof of Theorem 2, we observe that the number of vertices of  $H$  and the parameters  $d$  and  $k$  are polynomial in  $n + r$ , and therefore if we could solve MINIMUM SUBGRAPH OF MINIMUM DEGREE AT LEAST  $d$  in time  $2^{o(\mathbf{cwd}(H) \log d)} \cdot |V(H)|^{O(1)}$  then the RED-BLUE CDS could be solved in time  $2^{o(\mathbf{cwd}(I(G)) \log |V(G)|)} \cdot |V(G)|^{O(1)}$ . By Corollary 1, this cannot be done unless ETH fails.