CLIQUE-WIDTH IS NP-COMPLETE[*]

MICHAEL R. FELLOWS[†], FRANCES A. ROSAMOND[†], UDI ROTICS[‡], AND STEFAN SZEIDER[§]

Abstract. Clique-width is a graph parameter that measures in a certain sense the complexity of a graph. Hard graph problems (e.g., problems expressible in Monadic Second Order Logic with second-order quantification on vertex sets, that includes NP-hard problems such as 3-colorability) can be solved in polynomial time for graphs of bounded clique-width. We show that the clique-width of a given graph cannot be absolutely approximated in polynomial time unless P = NP. We also show that, given a graph G and an integer k, deciding whether the clique-width of G is at most k is NP-complete. This solves a problem that has been open since the introduction of clique-width in the early 1990s.

Key words. clique-width, NP-completeness, pathwidth, absolute approximation

AMS subject classifications. 68Q17, 05C75, 68Q42

1. Introduction.

Clique-width is a graph parameter that measures in a certain sense the complexity of a graph. This parameter is defined via a graph construction process where only a limited number of vertex labels are available; vertices that share the same label at a certain point of the construction process must be treated uniformly in subsequent steps. In particular, one can use the following four operations: the creation of a new vertex with label i, the vertex-disjoint union of already constructed labeled graphs, the insertion of all possible edges between vertices of specified labels, and the uniform relabeling of vertices. The clique-width $\mathrm{cwd}(G)$ of a graph G is the smallest number k of labels that suffice to construct G by means of these four operations. Such a construction of a graph can be represented by an algebraic term called a *k-expression*. More exact definitions are provided in Section 2.

This composition mechanism was first considered by Courcelle, Engelfriet, and Rozenberg [9, 10]. Clique-width can be considered to be more general than the popular graph parameter treewidth since there are graphs of constant clique-width but arbitrarily high treewidth (e.g., complete graphs), but graphs of bounded treewidth also have bounded clique-width [14, 7]. In recent years, clique-width has received much attention [4, 5, 3, 8, 6, 13, 11, 12, 14, 15, 7, 16, 19, 22, 26, 28, 27, 33].

In particular, the following result of Courcelle, Makowsky, and Rotics [11] makes the parameter clique-width attractive: any graph problem that can be expressed in Monadic Second Order Logic with second-order quantification on vertex sets (MSO_1) can be solved in linear time for graphs of clique-width bounded by some constant k; albeit the running time involves a constant which can be multiply exponential in k. A k-expression must be provided as input to the algorithm. Many NP-hard graph problems, e.g., 3-colorability, can be expressed in MSO_1. Seese [32] conjectured that if $\mathcal{C}$ is a class of graphs such that it is decidable for every property that can be expressed in MSO_1 whether there exists a graph in $\mathcal{C}$ that satisfies the property, then $\mathcal{C}$ is of

[*]An extended abstract of this paper appeared in the Proceedings of STOC 2006, 38th ACM Symposium on Theory of Computing, Seattle, Washington, USA, pp. 354–362, ACM Press, 2006.

[†]School of Electrical Engineering and Computer Science, University of Newcastle, Callaghan 2308 NSW, Australia (`[mike.fellows|frances.rosamond]@cs.newcastle.edu.au`).

[‡]School of Computer Science and Mathematics, Netanya Academic College, Netanya, Israel (`rotics@mars.netanya.ac.il`).

[§]Department of Computer Science, Durham University, Durham, England, UK (`stefan.szeider@durham.ac.uk`).

bounded clique-width. This conjecture can be considered as a kind of converse to the result of [11]. A slightly weaker form of Seese's Conjecture, where MSO_1 with parity predicates is considered, was recently shown by Courcelle and Oum [15].

A main limit for applications of the result of Courcelle, et al. [11] is that it is not known how to efficiently obtain k-expressions for graphs of clique-width k (except for graphs belonging to special classes [29]). In particular, the following question has been open since the introduction of clique-width in the early 1990s.

QUESTION 1. *Is it possible to compute the clique-width of a graph in polynomial time?*

Since the computation of treewidth is well-known to be NP-hard (Arnborg, Corneil, and Proskurowski [1]), it is obvious to assume that such a hardness result should also hold for the more general parameter clique-width. Despite considerable efforts, no hardness result for clique-width has until now been obtained. A main obstacle for giving a reduction from a known NP-hard problem is certainly the strong generative power of clique-width: even very few labels are sufficient to construct graphs of high connectivity; e.g., two labels are sufficient to construct complete graphs of arbitrary size (see below).

In the present paper we answer Question 1 negatively as follows.

THEOREM 1. *Let ε be a constant with $0 \leq \varepsilon < 1$. The clique-width of graphs with n vertices of degree greater than 2 cannot be approximated by a polynomial-time algorithm with an absolute error guarantee of n^ε unless* $\mathrm{P} = \mathrm{NP}$.

In particular, there is no polynomial-time absolute approximation algorithm for clique-width unless $\mathrm{P} = \mathrm{NP}$.

THEOREM 2. cwd-RECOGNITION *is* NP*-complete.*

Here, cwd-RECOGNITION refers to the following decision problem: given a graph G and an integer k, is the clique-width of G at most k? We shall use a similar terminology for other graph parameters such as pathwidth (pwd).

Our proofs rely ultimately on a reduction from pwd-RECOGNITION. However, a direct reduction, where an instance (G, k) of pwd-RECOGNITION is reduced to an instance (G', k') of cwd-RECOGNITION, seems to be difficult to obtain, as the combinatorics for such a construction would be obliged to be "tight." We base our reduction on the following stronger result of Bodlaender, Gilbert, Hafsteinsson, and Kloks [2]: It is NP-hard to approximate pathwidth within an absolute error guarantee. This stronger basis allows the reduction to be lax, making the combinatorial effort somehow easier than it would be for a direct reduction.

Obtaining the hardness for the exact solution of one problem from the non-approximability of another problem is the key for our success with respect to Question 1. We believe that this approach could be useful in many situations, possibly also in parameterized complexity.

1.1. Related Work. Corneil, Habib, Lanlignel, Reed, and Rotics [6] show that graphs of clique-width at most 3 can be recognized in polynomial time. Oum and Seymour [31] present an algorithm that, for fixed k, computes $(2^{3k+2} - 1)$-expressions for n-vertex graphs of clique-width at most k in time $O(n^9 \log n)$. This result renders the notion "class of bounded clique-width" feasible in much the same way that "class of bounded treewidth" is feasible. The algorithm of Oum and Seymour computes first an approximate "rank-decomposition" (a decomposition associated with the graph invariant rank-width, for definitions see [31, 30]) and then converts the rank-decomposition of width k into a $(2^{k-1} + 1)$-expressions. Oum [30] gives improved algorithms for rank-width approximation that also give rise to improved algorithms for clique-width (however, still with an approximation error that is exponential in the clique-width).

The graph parameter "NLC-width" introduced by Wanke [34] is defined similarly to clique-width using a single type of operation that combines disjoint union and insertion of edges. Gurski and Wanke [20] show that NLC-width-RECOGNITION is NP-hard. Since NLC-width and clique-width can differ by a factor of 2 (see Johansson [23]), non-approximability with an absolute error guarantee (or NP-hardness of recognition) for one of the two parameters does not imply a similar result for the other parameter.

2. Definitions and preliminaries.

Unless otherwise stated, all graphs considered in this paper are finite, undirected, and simple.

2.1. Clique-width.

Let k be a positive integer. A *k-graph* is a graph whose vertices are labeled by integers from $\{1,\dots,k\}$. We consider an arbitrary graph as a k-graph with all vertices labeled by 1. We call the k-graph consisting of exactly one vertex v (say, labeled by $i \in \{1,\dots,k\}$) an *initial k-graph* and denote it by $i(v)$.

The *clique-width* $\mathrm{cwd}(G)$ of a graph G is the smallest integer k such that G can be constructed from initial k-graphs by means of repeated application of the following three operations.

1. *Disjoint union* (denoted by $\oplus$);
2. *Relabeling*: changing all labels i to j (denoted by $\rho_{i\to j}$);
3. *Edge insertion*: connecting all vertices labeled by i with all vertices labeled by j, $i \neq j$ (denoted by $\eta_{i,j}$ or $\eta_{j,i}$); already existing edges are not doubled.

A construction of a k-graph using the above operations can be represented by an algebraic term composed of $\oplus$, $\rho_{i\to j}$, and $\eta_{i,j}$, ($i, j \in \{1,\dots,k\}$, and $i \neq j$). Such a term is called a *cwd-expression* defining G. A *k-expression* is a cwd-expression in which at most k different labels occur. Thus, the clique-width of a graph G is the smallest integer k such that G can be defined by a k-expression.

For example, the complete graph K_4 on the vertices u, v, w, x is defined by the cwd-expression

$$\rho_{2\to 1}(\eta_{1,2}(\rho_{2\to 1}(\eta_{1,2}(\rho_{2\to 1}(\eta_{1,2}(2(u) \oplus 1(v))) \oplus 2(w))) \oplus 2(x))).$$

Hence $\mathrm{cwd}(K_4) \leq 2$. In general, every complete graph on $n \geq 2$ vertices has clique-width exactly 2.

2.2. Directed clique-width.

One can use a k-expression to construct a directed graph, interpreting $\eta_{i,j}$ as the operation that inserts directed edges that are oriented from vertices labeled i to vertices labeled j. Courcelle and Olariu [14] define the clique-width of a directed graph D as the smallest integer k such that D can be constructed by a k-expression. Theorems 1 and 2 carry over to directed graphs; this follows from the following considerations.

Let G be an undirected graph, and let D be the directed graph obtained from G by replacing every undirected edge uv of G with two directed edges (u, v) and (v, u). It is easy to see that $\mathrm{cwd}(G) = \mathrm{cwd}(D)$ since from any k-expression for G we can build a k-expression for D by adding an $\eta_{i,j}$-operation immediately after any $\eta_{j,i}$-operation. Thus, Theorems 1 and 2 imply their counterparts for directed graphs.

2.3. Linear clique-width.

A cwd-expression is *linear* if whenever two k-graphs are put together by disjoint union, at least one of the two graphs is an initial k-graph (in contrast to non-linear cwd-expressions where both k-graphs can be arbitrarily large). In other words, a linear k-expression encodes the composition of a graph

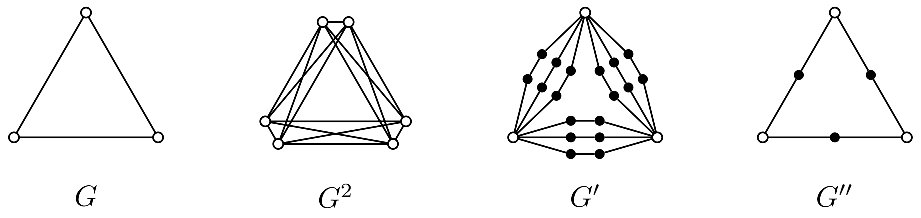

FIG. 3.1. *Illustration for Constructions 1, 2, and 3.*

starting with a single vertex to which one adds successively further vertices, one after the other, with interleaved operations of relabeling and edge insertion. The parse trees of linear k-expressions are path-like; hence one can consider the relation between linear clique-width and clique-width as analogous to the relation between pathwidth and treewidth. The *linear clique-width* lin-cwd(G) of a graph G is the smallest integer k such that G can be defined by a linear k-expression.

Every complete graph on $n \geq 2$ vertices has linear clique-width exactly 2 (observe that the cwd-expression for K_4 as given above is linear). However, the difference between the clique-width and the linear clique-width of a graph can be arbitrarily large. Let T_h denote a complete ternary tree of height h and let T'_h denote the graph obtained from T_h by means of Construction 2 as defined below. In [17] we show that the clique-width of T'_h is at most 4, but the linear clique-width of T'_h is at least $h-1$.

2.4. Treewidth and pathwidth. Let T be a tree and χ a labeling of the vertices of T by sets of vertices of a graph G. The pair (T,χ) is a *tree decomposition* of G if (i) every vertex of G belongs to $\chi(t)$ for some vertex t of T; (ii) for every edge vw of G there is some vertex t of T with $v, w \in \chi(t)$; (iii) for any vertices t_1, t_2, t_3 of T, if t_2 lies on a path from t_1 to t_3, then $\chi(t_1) \cap \chi(t_3) \subseteq \chi(t_2)$. The *width* of (T,χ) is the maximum $|\chi(t)|-1$ over all vertices t of T. The *treewidth* twd(G) of G is the minimum width over all tree-decompositions of G. The *pathwidth* pwd(G) of G is the minimum width over all tree-decompositions (T,χ) of G where T is a path.

2.5. Cobipartite graphs. A graph is *cobipartite* if it is the complement of a bipartite graph. That is, the vertex set of a cobipartite graph G can be partitioned into two sets A and B such that each of them induces in G a complete subgraph. Several graph parameters, including treewidth and pathwidth, agree on cobipartite graphs (see, e.g., Fomin, Heggernes, and Telle [18]).

3. Proof of the main results.

In what follows, let α denote an integer-valued graph parameter. We consider the following decision problem.

α-RECOGNITION
Instance: A graph G and a positive integer k.
Question: Is $\alpha(G)$ at most k?

Arnborg et al. [1] have shown that pwd-RECOGNITION is NP-complete, even for cobipartite graphs.

The following construction is due to Bodlaender, Gilbert, Hafsteinsson, and Kloks [2] (this and other constructions defined below are illustrated in Fig. 3.1).

CONSTRUCTION 1. Given a graph G, and an integer $q \geq 1$, we obtain a graph G^q by replacing each vertex v of G by q vertices $v_1, \ldots, v_q$, and by joining two vertices v_i, w_j by an edge if either $v = w$ and $i \neq j$, or $v \neq w$ and vw is an edge of G.

Note that if G is cobipartite then so is G^q (as pointed out by Karpinski and Wirtgen [24]). Bodlaender, et al. [2] show the following equation (the proof stated in [2] for treewidth applies literally to pathwidth as well).

$$\operatorname{pwd}(G^q) = q(\operatorname{pwd}(G) + 1) - 1. \tag{3.1}$$

Hence, if we apply an absolute approximation algorithm for pathwidth to G^q for q large enough, then we can use equation (3.1) to calculate the exact pathwidth of G. This idea provides the basis for the next lemma.

LEMMA 1. *Assume that there is a constant c such that $|\alpha(G) - \operatorname{pwd}(G)| \le c$ holds for every cobipartite graph G with minimum degree at least 3. Then the following statements are true.*

1. *Unless P = NP there is no constant ε, $0 \le \varepsilon < 1$, such that for graphs G with n vertices and minimum degree at least 3, $\alpha(G)$ can be approximated in polynomial-time with an absolute error guarantee of n^ε.*
2. *α-RECOGNITION is NP-hard.*

Proof. Part 1. Let ε be a fixed constant with $0 \le \varepsilon < 1$. Assume that there exists a polynomial-time algorithm $\mathcal{A}$ that outputs for a given graph G with n vertices and minimum degree at least 3 an integer $\mathcal{A}(G)$ such that $|\mathcal{A}(G) - \alpha(G)| \le n^\varepsilon$. Thus, we have $|\mathcal{A}(G) - \operatorname{pwd}(G)| \le n^\varepsilon + c$. We choose a constant $d \ge 2$ such that $\varepsilon < d/(d+3)$. Let G be a given cobipartite graph with $n \ge 2$ vertices where n is large enough to satisfy $n^{(d+2)\varepsilon} + c \le n^{(d+3)\varepsilon}$. For $q = n^{d+1}$ we form the graph G^q using Construction 1 (observe that since d is a constant, G^q can be formed in polynomial time). Note that G^q is cobipartite, of minimum degree at least $q - 1 \ge 3$, and the number of vertices of G^q is exactly n^{d+2}. We apply algorithm $\mathcal{A}$ to G^q and get $|\mathcal{A}(G^q) - \alpha(G^q)| \le n^{(d+2)\varepsilon}$. The assumption of the lemma gives $|\mathcal{A}(G^q) - \operatorname{pwd}(G^q)| \le n^{(d+2)\varepsilon} + c \le n^{(d+3)\varepsilon} < n^d$. Using equation (3.1) we get $|\mathcal{A}(G^q) - q(\operatorname{pwd}(G) + 1) + 1| < n^d$, hence $|[(\mathcal{A}(G^q) + 1)/q - 1] - \operatorname{pwd}(G)| < 1/n \le 1/2$. Hence we can use algorithm $\mathcal{A}$ to compute the exact pathwidth of G in polynomial time. By the aforementioned theorem of Arnborg et al., this is not possible unless P = NP.

Part 2. We reduce pwd-RECOGNITION to α-RECOGNITION. Let (G, k) be an instance of pwd-RECOGNITION. Since pwd-RECOGNITION is NP-hard for cobipartite graphs, we may assume that G is cobipartite. We obtain in polynomial time an instance (G^*, k^*) of α-RECOGNITION, putting $G^* = G^{2c+3}$ and $k^* = (2c+3)(k+1) + c - 1$. Observe that G^* is cobipartite and has minimum degree at least 3. We show that (G, k) is a yes-instance of pwd-RECOGNITION if and only if (G^*, k^*) is a yes-instance of α-RECOGNITION; that is, $\operatorname{pwd}(G) \le k$ if and only if $\alpha(G^*) \le k^*$. First assume $\operatorname{pwd}(G) \le k$. Now by equation (3.1), $\alpha(G^*) \le \operatorname{pwd}(G^*) + c = (2c+3)(\operatorname{pwd}(G) + 1) - 1 + c \le (2c+3)(k+1) + c - 1 = k^*$, thus $\alpha(G^*) \le k^*$ follows. Conversely, assume $\alpha(G^*) \le k^*$. We have $(2c+3)(\operatorname{pwd}(G) + 1) - 1 = \operatorname{pwd}(G^*) \le \alpha(G^*) + c \le k^* + c = (2c+3)(k+1) + c - 1 + c$. Hence, $\operatorname{pwd}(G) \le k + 2c/(2c+3)$. Since $\operatorname{pwd}(G)$ and k are integers, $\operatorname{pwd}(G) \le k$ follows. ☐

We shall use the following two constructions.

CONSTRUCTION 2. Let G denote a graph with $n \ge 2$ vertices. We obtain a graph G' from G by replacing each edge uv of G by three internally disjoint paths $u - x_i - y_i - v$, $i = 1, 2, 3$, of length 3 (the vertices x_i, y_i are new vertices); we call such paths *bridges*.

CONSTRUCTION 3. Let G denote a graph with $n \ge 2$ vertices. We obtain a graph G'' from G by replacing each edge uv of G by a path $u - s_{u,v} - v$, where $s_{u,v}$ is a new vertex.

In Section 5 we show that the following inequalities hold.

$$\text{pwd}(G) \leq \text{lin-cwd}(G') \leq \text{pwd}(G) + 4. \tag{3.2}$$

In view of Lemma 1 this already implies the NP-hardness of lin-cwd-RECOGNITION (see the end of Section 5). However, the gap between cwd(G) and lin-cwd(G) can be arbitrarily large [21, 17]. Hence the hardness result for linear clique-width does not extend directly to clique-width. Fortunately, we can bound the gap between cwd(G') and lin-cwd(G') by a small constant if G' is obtained from a cobipartite graph G of minimum degree at least 2 by means of Construction 2. In particular, for such graphs G' we establish the following inequalities.

$$\text{cwd}(G') \leq \text{lin-cwd}(G') \leq \text{cwd}(G') + 18. \tag{3.3}$$

We obtain the non-trivial part of inequality (3.3) by means of Construction 3. We show by Lemma 7, Theorem 5, and Lemma 8, respectively, that for every cobipartite graph G of minimum degree at least 2 we have

$$\text{lin-cwd}(G') \leq \text{lin-cwd}(G'') + 9 \leq \text{cwd}(G'') + 15 \leq \text{cwd}(G') + 18. \tag{3.4}$$

The hardest task for showing (3.4) is to bound the linear clique-width of G'' in terms of the clique-width of G'' plus a small constant; this is established in Theorem 5. There we start with an arbitrary k-expression defining G'' and modify this k-expression in several steps. In each step we introduce only a small number of new labels, and finally we are left with a *linear* $(k+6)$-expression defining G''.

Consider now the graph parameter

$$\alpha(G) = \text{cwd}(G');$$

i.e., $\alpha(G)$ is the clique-width of the graph G' obtained from G by Construction 2. Inequalities (3.2) and (3.3) yield $|\alpha(G) - \text{pwd}(G)| \leq 18$, hence the assumption of Lemma 1 is met. It is now easy to establish Theorems 1 and 2 as follows.

Proof of Theorem 1. Assume that for a constant ε, $0 \leq \varepsilon < 1$, there exists a polynomial-time algorithm $\mathcal{A}$ that outputs for a given graph G with n vertices of degree at least 3 an integer $\mathcal{A}(G)$ with $|\mathcal{A}(G) - \text{cwd}(G)| \leq n^{\varepsilon}$. For a graph G with n vertices and minimum degree at least 3, G' has exactly n vertices of degree at least 3; applying $\mathcal{A}$ to G' gives now $|\mathcal{A}(G') - \text{cwd}(G')| = |\mathcal{A}(G') - \alpha(G)| \leq n^{\varepsilon}$. Hence, by the first part of Lemma 1 such algorithm $\mathcal{A}$ cannot exist unless P = NP. A similar reasoning applies if the approximation error is bounded by some fixed constant. □

Proof of Theorem 2. The second part of Lemma 1 implies that α-RECOGNITION is NP-hard. We reduce α-RECOGNITION to cwd-RECOGNITION by taking for an instance (G, k) of the former problem the instance (G', k) of the latter problem; obviously $\alpha(G) \leq k$ if and only if $\text{cwd}(G') \leq k$. Thus cwd-RECOGNITION is NP-hard as well. The problem is in NP since, given a graph G, we can guess a k-expression and check in polynomial time whether it is indeed a k-expression defining G. □

4. Further notation on clique-width. In this section we introduce further (and more technical) notations for cwd-expressions that we will use in the proofs below.

A vertex v of a labeled graph G is a *singleton* if there is no other vertex of G that has the same label as v. For a cwd-expression t, we denote by $\textbf{val}(t)$ the labeled graph defined by t. We denote a cwd-expression which uses at most k labels as a k-expression; for convenience we assume that the k labels are the integers $1, \ldots, k$. For a labeled graph G we denote by $\mathbf{labels}(G)$ the number of labels used in G.

For a cwd-expression t defining a graph G, we denote by $\mathbf{tree}(t)$ the parse tree constructed from t in the usual way. Fig. 4.1 shows the tree for the cwd-expression for K_4 given in Section 2.1. The leaves of $\mathbf{tree}(t)$ are the vertices of G with their initial

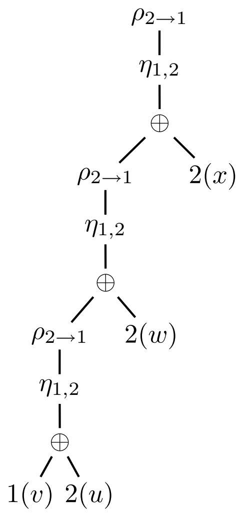

FIG. 4.1. *Parse tree.*

labels, and the internal nodes of $\mathbf{tree}(t)$ correspond to the operations of t and can be either binary, corresponding to $\oplus$, or unary, corresponding to η or ρ. For a node a of $\mathbf{tree}(t)$, we denote by $\mathbf{tree}(t)\langle a\rangle$ the subtree of $\mathbf{tree}(t)$ rooted at a. We denote by $t\langle a\rangle$ the cwd-expression corresponding to $\mathbf{tree}(t)\langle a\rangle$; i.e., $\mathbf{tree}(t)\langle a\rangle = \mathbf{tree}(t\langle a\rangle)$. Note that in $t\langle a\rangle$, and similarly in $\mathbf{tree}(t\langle a\rangle)$, we assume that the operation at a is already established (this operation is the leading symbol of $t\langle a\rangle$).

For a vertex x of $\mathbf{val}(t\langle a\rangle)$, we say that x is *dead at* a, or *dead at* $\mathbf{val}(t\langle a\rangle)$, if all the edges incident to x in $\mathbf{val}(t)$ are included in $\mathbf{val}(t\langle a\rangle)$. Otherwise we say that x is *active at* a, or *active at* $\mathbf{val}(t\langle a\rangle)$. We say that label ℓ is *dead* in t if it is not involved in any η-operation in t. In other words, ℓ is dead in t if there is no η-operation in t of the form $\eta_{\ell,\ell'}$ for any label ℓ'.

Let a be a $\oplus$-operation of a cwd-expression t. If z is a vertex of $\mathbf{val}(t\langle a\rangle)$ and has label ℓ in $\mathbf{val}(t\langle a\rangle)$ we say that *z occurs at a with label ℓ*. Let b and c be the left and right children of a, respectively. We say that vertex x occurs on the *left (right) side* of a if it occurs at b (c). We say that a is a *1-$\oplus$-operation* if there is exactly one vertex occurring on the left side of a or there is exactly one vertex occurring on the right side of a. We say that a is a *(>1)-$\oplus$-operation* if it is not a 1-$\oplus$-operation. Thus, t is a linear cwd-expression if all $\oplus$-operations in t are 1-$\oplus$-operations.

Let G' and G'' be the graphs obtained from G by means of Constructions 2 and 3, respectively. We call the vertices of G' and G'' which are also vertices of G *regular vertices*. We call the vertices of G' and G'' which are not vertices of G *special vertices*.

5. Comparing pathwidth and linear clique-width.

This section is devoted to showing that the pathwidth of G and the linear clique-width of G' differ at most by 4.

We shall use a characterization of pathwidth by means of graph layouts. A (linear) *layout* (or arrangement) of a graph $G = (V, E)$ with n vertices is a 1-to-1 map $\varphi : V \to \{1, \ldots, n\}$. For a layout φ of G and $i \in \{1, \ldots, n\}$ we define four sets of vertices: $L_G(i, \varphi)$ is the set of vertices $u \in V$ with $\varphi(u) \le i$; we put $R_G(i, \varphi) = V \setminus L_G(i, \varphi)$. $L^*_G(i, \varphi)$ is the set of vertices of $L_G(\varphi, i)$ that have a neighbor in $R_G(i, \varphi)$; symmetrically, $R^*_G(i, \varphi)$ is the set of vertices of $R_G(i, \varphi)$ that have a neighbor in $L_G(i, \varphi)$. The *in-degree* and the *out-degree* of φ is defined as $\max_{i=1}^{n-1} |R^*_G(i, \varphi)|$ and $\max_{i=1}^{n-1} |L^*_G(i, \varphi)|$, respectively. The *vertex separation number* $\mathrm{vsn}(G)$ of G is defined as the smallest in-degree over all layouts of G (which equals the smallest out-degree over all layouts of G). It is well-known that pathwidth equals the vertex separation number [25]. Using the notion of vertex separation number, it is easy to see that for every graph G we have

$$\text{lin-cwd}(G) \le \mathrm{pwd}(G) + 2. \tag{5.1}$$

Namely, consider a layout φ of G with in-degree k. A linear $(k+2)$-expression defining G can be obtained from φ as follows. We introduce the vertices according to φ one after the other. At step i of the construction process we introduce vertex $\varphi(i)$ with label $k+1$; at this stage all vertices in $L^*_G(i-1, \varphi)$ have distinct labels from $\{1, \ldots, k\}$. We can connect $\varphi(i)$ with its neighbors in $L^*_G(i-1, \varphi)$ using separate η-operations. Thereafter, we change the labels of vertices in $L^*_G(i-1, \varphi) \setminus L^*_G(i, \varphi)$ to the dead label $k+2$. Since $L^*_G(i, \varphi) \le k$, at least one label $j \in \{1, \ldots, k\}$ is now available; we change the label of $\varphi(i)$ to j and continue. This process yields a linear $(k+2)$-expression defining G.

For the remainder of this section G denotes a fixed graph with $n \ge 2$ vertices and G' denotes the graph obtained from G by means of Construction 2.

LEMMA 2. *If G admits a layout with in-degree k, then the linear clique-width of G' is at most $k+4$. Hence* $\text{lin-cwd}(G') \le \mathrm{pwd}(G) + 4$.

Proof. Assume $\mathrm{pwd}(G) = k$. Hence there exists a layout φ of G with in-degree k. For $i = 1, \ldots, n$ let Γ_i denote the set of vertices of G' that belong to $L_G(i, \varphi)$ or are of distance at most 2 apart from $L_G(i, \varphi)$. Thus, if at least one end of a bridge b belongs to $L_G(i, \varphi)$, then both internal vertices of b belong to Γ_i. Let Δ_i denote the subset of Γ_i consisting of vertices that are adjacent in G' with vertices outside of Γ_i. Furthermore, let G'_i denote the subgraph of G' induced by the set Γ_i.

We inductively obtain linear $(k+4)$-expressions t_i defining G'_i, $i = 1, \ldots, n$, such that the labeling of $\mathbf{val}(t_i)$ satisfies the following conditions.

1. vertices in $\Gamma_i \setminus \Delta_i$ are labeled by 1;
2. vertices in Δ_i are labeled by integers from $5 \ldots, k+4$;
3. two vertices of Δ_i share the same label if and only if both vertices have a common neighbor in G'.

Let $f : R^*_G(1, \varphi) \to \{5, \ldots, k+4\}$ be an injective map (such map exists since $|R^*_G(1, \varphi)| \le k$). The expression t_1 is obtained as follows.

1. We start with the term $2(u)$ which introduces the vertex $u = \varphi^{-1}(1)$ with label 2.

2. For every pair x, y of vertices that lie on a bridge between u and some vertex $v \in R^*_G(1, \varphi)$ we add the following sequence of operations.

2.1. A 1-$\oplus$-operation that adds vertex x with label 3.

2.2. An $\eta_{2,3}$-operation that connects u and x.

2.3. A 1-$\oplus$-operation that adds y with label 4.

2.4. An $\eta_{3,4}$-operation that connects x and y.

2.5. A $\rho_{3\to 1}$-operation that gives x the dead label 1.

2.6. A $\rho_{4\to f(v)}$ operation that gives y the label $f(v)$.

3. Finally, we add a $\rho_{2\to 1}$-operation that gives u the dead label 1.

The linear $(k{+}4)$-expression t_1 defines G'_1 and the claimed properties are evidently satisfied. Now assume that we have already a $(k+4)$-expression t_{i-1} defining G'_{i-1} for some $i \in \{2,\ldots,n\}$ such that $\mathbf{val}(t_{i-1})$ satisfies the claimed properties.

For vertices $v \in R^*_G(i,\varphi)$ let $\Delta_{i-1}(v)$ denote the set of vertices in Δ_{i-1} that are adjacent to v in G'. Let $u = \varphi^{-1}(i)$. By assumption, there is an injective map $f' : R^*_G(i-1,\varphi) \to \{5,\ldots,k+4\}$ such that all vertices in $\Delta_{i-1}(u)$ have the same label $f'(u)$ in $\mathbf{val}(t_{i-1})$, and no other vertex of $\mathbf{val}(t_{i-1})$ is labeled with $f'(u)$. Since $|R^*_G(i,\varphi)| \le k$, we can define an injective map $f : R^*_G(i,\varphi) \to \{5,\ldots,k+4\}$ with $f(v) = f(v)'$ for $v \in R^*_G(i-1,\varphi) \cap R^*_G(i,\varphi)$.

We extend t_{i-1} to a $(k+4)$-expression t_i defining G'_i, adding the following operations immediately above the root of $\mathbf{tree}(t_{i-1})$.

1. Add a 1-$\oplus$-operation that introduces the vertex u with label 2 (u is now the only vertex labeled with 2).

2. Add an $\eta_{f'(u),2}$-operation that connects all vertices in $\Delta_{i-1}(u)$ with u.

3. Add a $\rho_{f'(u)\to 1}$-operation that gives all vertices in $\Delta_{i-1}(u)$ the dead label 1.

4. As above, for every pair x, y of vertices that lie on a bridge between u and some $v \in R^*_G(1,\varphi)$ we add the sequence of operations 2.1–2.6 given above.

5. Finally, we add a $\rho_{2\to 1}$-operation that gives u the dead label 1.

It is straightforward to verify that the obtained linear $(k+4)$-expression defines G'_i and that the labeling of $\mathbf{val}(t_i)$ satisfies the claimed properties.

Since $G'_n = G'$, it follows that the linear clique-width of G' is at most $k+4$. □

The next lemma will allow us to bound the pathwidth of G in terms of the linear clique-width of G', a result inverse to Lemma 2. To this end let us fix a linear k-expression t defining G'. Since t is linear it gives raise to a sequence $t_1,\ldots,t_s$ of linear k-expressions such that t_1 defines an initial k-graph, $t_s = t$, and for each $i \in \{2,\ldots,s\}$, t_i is obtained from t_{i-1} by adding a ρ-operation, a η-operation, or 1-$\oplus$-operation that introduces a new vertex. For every edge e of G' let $j(e) := \min\{\, 1 \le j \le s : e \in \mathbf{val}(t_j)\,\}$. We call a bridge $u - x - y - v$ of G' *well-behaved* (relative to t) if u is a singleton in $\mathbf{val}(t_{j(ux)})$ and v is a singleton in $\mathbf{val}(t_{j(yv)})$.

Lemma 3. *At least one of any three parallel bridges of G' is well-behaved.*

Proof. For an edge uv of G let b_1, b_2, b_3 denote the three corresponding bridges of G', where b_i is the bridge $u - x_i - y_i - v$, $i = 1,2,3$. For $i = 1,2,3$ we put $\alpha_i = \max(j(ux_i), j(y_iv))$.

Claim A: $j(ux_i)$ and $j(y_iv)$ must be distinct for $i = 1,2,3$. Otherwise, either u would have the same label as y_i or the same label as v in $\mathbf{val}(t_{j(ux_i)})$. In the first case, the addition of the edge y_iv causes the addition of the edge uv. In the second case, the addition of the edge y_iv causes the addition of the edge y_iu. However, neither uv nor y_iu is present in G'. Hence Claim A is shown.

Claim B: if $j(ux_i) < j(y_iv)$, then u is singleton in $\mathbf{val}(t_{j(ux_i)})$ for $i = 1,2,3$. Assume to the contrary that there is a vertex $w \ne u$ in $\mathbf{val}(t_{j(ux_i)})$ which shares the label with u. It follows that G' contains the edge wx_i, hence $w = y_i$. This, however, implies that $\mathbf{val}(t_{j(y_iv)})$ contains the edge uv, a contradiction. Hence Claim B is shown.

Now we proceed with the proof of the lemma. We consider two cases.

Case 1: $|\{\alpha_1,\alpha_2,\alpha_3\}| \le 2$. We assume, w.l.o.g., $\alpha_1 = \alpha_2 = j(y_1v)$. Clearly $\alpha_2 = j(y_2v)$, since otherwise, if $\alpha_2 = j(ux_2)$, then some of the edges uy_1, uv were present in G'. Let w be a vertex of $\mathbf{val}(t_{j(y_1v)})$ that shares the label with v. It follows that G' contains the edges wy_1 and wy_2; hence $w = v$. Thus v is a singleton in $\mathbf{val}(t_{j(y_1v)})$. Since $j(ux_1) < j(y_1v)$, it follows from Claim B that u is a singleton in $\mathbf{val}(t_{j(ux_1)})$. Hence the bridge b_1 is well-behaved.

Case 2: $|\{\alpha_1, \alpha_2, \alpha_3\}| = 3$. We assume, w.l.o.g., that $j(y_1v) = \alpha_1 < \alpha_2 < \alpha_3$.

Subcase 2a: $j(y_2v) > j(y_1v)$ or $j(y_3, v) > j(y_1v)$. W.l.o.g., $j(y_2v) > j(y_1v)$. Similarly as above we conclude that for any vertex w of $\mathbf{val}(t_{j(y_1v)})$ that shares the label with v, the edges y_1w and y_2w are present in $\mathbf{val}(t_{j(y_1v)})$ and $\mathbf{val}(t_{j(y_2v)})$, respectively. Hence $w = v$, and so v is a singleton in $\mathbf{val}(t_{j(y_1v)})$. Furthermore, since $j(ux_1) < j(y_1v)$, it follows by Claim B that u is a singleton in $\mathbf{val}(t_{j(ux_1)})$. Hence the bridge b_1 is well-behaved.

Subcase 2b: $j(y_2v) \le j(y_1v)$ and $j(y_3, v) \le j(y_1v)$. It follows that $\alpha_2 = j(ux_2)$ and $\alpha_3 = j(ux_3)$. We show that u is a singleton in $\mathbf{val}(t_{j(ux_2)})$. Let w be a vertex of $\mathbf{val}(t_{j(ux_2)})$ that shares the label with u. Consequently, the edges wx_2 and wx_3 are present in $\mathbf{val}(t_{j(ux_2)})$ and $\mathbf{val}(t_{j(ux_3)})$, respectively. Thus $u = w$ and so u is indeed a singleton in $\mathbf{val}(t_{j(ux_2)})$. Using a symmetrical version of Claim B, we conclude from $j(y_2v) < j(ux_2)$ that v is a singleton in $\mathbf{val}(t_{j(y_2v)})$. Hence the bridge b_2 is well-behaved. ∎

LEMMA 4. *If G' has linear clique-width k, then G admits a layout of in-degree at most k. Hence* $\mathrm{pwd}(G) \le \text{lin-cwd}(G')$.

Proof. Let k be the number of labels used in t. For a vertex v of G let $\beta(v)$ denote the smallest integer in $\{1, \ldots, s\}$ such that v is not a singleton of $\mathbf{val}(t_{\beta(v)})$. Note that $\beta(v)$ is defined for every vertex v of G, since we assume that G has more than one vertex and all vertices of $\mathbf{val}(t)$ have label 1. Note also that if $\beta(v) = \beta(v') = j$ holds for two vertices v, v' of G, then v and v' have the same label in $\mathbf{val}(t_j)$, but no other vertex in $\mathbf{val}(t_j)$ shares its label with v and v' (either v and v' are singletons in $\mathbf{val}(t_{j-1})$ and one of the two vertices is relabeled with the other's label in $\mathbf{val}(t_j)$, or one of the two vertices is a singleton in $\mathbf{val}(t_{j-1})$ and the other vertex is introduced in $\mathbf{val}(t_j)$ with the same label). Let φ be a layout of G satisfying $\varphi(v) < \varphi(v')$ whenever $\beta(v) < \beta(v')$.

We show that the in-degree of the layout φ is at most k. Choose $i \in \{1, \ldots, n-1\}$ arbitrarily. We show that $|R^*_G(i, \varphi)| \le k$. Let $w = \varphi^{-1}(i)$, $j = \beta(w)$, and consider the k-graph $\mathbf{val}(t_j)$. By construction, the vertices in $L_G(i, \varphi)$ are not singletons of $\mathbf{val}(t_j)$. We assign to every vertex $v \in R^*_G(i, \varphi)$ a label $f(v) \in \{1, \ldots, k\}$ as follows (it will turn out that f is an injective map). Choose arbitrarily a vertex $v \in R^*_G(i, \varphi)$. By definition, v is in G adjacent to a vertex $u \in L_G(i, \varphi)$. Thus u and v are joined by three parallel bridges in G'. By Lemma 3, at least one of the bridges between u and v, say $b = (u, x_v, y_v, v)$, is well-behaved in t. For vertices z of $\mathbf{val}(t_j)$ let $\ell(z)$ denote the label of z in $\mathbf{val}(t_j)$. We put

$$f(v) = \begin{cases} \ell(v) & \text{if } v \in \mathbf{val}(t_j); & \text{(c1)} \\ \ell(y_v) & \text{if } v \notin \mathbf{val}(t_j) \text{ and } y_v \in \mathbf{val}(t_j); & \text{(c2)} \\ \ell(x_v) & \text{if } v, y_v \notin \mathbf{val}(t_j). & \text{(c3)} \end{cases}$$

Since u is not a singleton in $\mathbf{val}(t_j)$, the edge ux_v must already be present in $\mathbf{val}(t_j)$ as the bridge $u - x_v - y_v - v$ is well-behaved. Consequently the above case distinction is exhaustive. We split the set $R^*_G(i, \varphi)$ into sets C_1, C_2, and C_3, such that a vertex v belongs to C_i if $f(v)$ is assigned by means of the above case (ci). We further split C_1 into sets $C_1^=$ and $C_1^<$ such that $v \in C_1$ belongs to $C_1^=$ if $\beta(w) = \beta(v)$ and v belongs to $C_1^<$ if $\beta(w) < \beta(v)$.

To show that f is an injective map, suppose to the contrary that $f(v) = f(v')$ for two distinct vertices $v, v' \in R^*_G(i,\varphi)$. Since the vertices of $C_1^<$ are singletons in $\mathbf{val}(t_j)$, $v, v' \notin C_1^<$ follows. For any $v \in C_3$, the vertex x_v is a singleton in $\mathbf{val}(t_j)$ since the edge $x_v y_v$ is still missing, hence $v, v' \notin C_3$. Furthermore, v and v' cannot both belong to $C_1^=$ since then both would share the label with w in $\mathbf{val}(t_j)$, but as seen above, any $v \in C_1^=$ shares its label only with w. If $v \in C_1^=$ and $v' \in C_2$, then when the edge $y_{v'}v'$ is established, also the edge vv' is established, a contradiction, since vv' is not an edge of G'. Hence we are left with the case $v, v' \in C_2$. Thus $f(v)$ is the label of y_v and $f(v')$ is the label of $y_{v'}$. The edges $y_v v, y_{v'}v'$ are not yet present in $\mathbf{val}(t_j)$ since the vertices v, v' are not yet present in $\mathbf{val}(t_j)$ either. If at a further step the edge $y_v v$ is added, also the edge $y_v v'$ is added, but G' does not contain the edge $y_v v'$, a contradiction. Thus $f : R^*_G(i,\varphi) \to \{1,\ldots,k\}$ is indeed an injective map, and so $|R^*_G(i,\varphi)| \le k$ follows. Hence we have shown that $\mathrm{pwd}(G) = \mathrm{vsn}(G) \le k$. ☐

Having established inequality (3.2), we can use a similar reasoning as outlined in Section 3 (however, setting $\alpha(G) = \text{lin-cwd}(G')$) to establish the following results.

THEOREM 3. *Let ε be a constant with $0 \le \varepsilon < 1$. The linear clique-width of graphs with n vertices of degree greater than 2 cannot be approximated by a polynomial-time algorithm with an absolute error guarantee of n^ε unless* P = NP.

In particular, there is no polynomial-time absolute approximation algorithm for linear clique-width unless P = NP.

THEOREM 4. lin-cwd-RECOGNITION *is* NP-*complete.*

6. Comparing Construction 2 and Construction 3.

For this section let G denote a graph with minimum degree at least 2. We show that the clique-width of G'' is bounded by the clique-width of G' plus a small constant, and that the converse is true for linear clique-width.

6.1. From G'' to G'.

PROPERTY 1. Let t be a linear cwd-expression defining G''. We say that t has *Property 1* if the following two conditions are satisfied.

Condition 1.1: Every η-operation a in t establishes some edge which is not established by another η-operation above a in $\mathbf{tree}(t)$.

Condition 1.2: For every two regular vertices x and y there is no node a in $\mathbf{tree}(t)$ such that x and y are active at a and have the same label at a.

LEMMA 5. *Let t be a linear k-expression defining G''. Then there exists a linear $(k+2)$-expression defining G'' which has Property 1.*

Proof. Let t be a linear k-expression defining G''. Since we can remove η-operations that invalidate Condition 1.1 from t, we may assume that all η-operations in t satisfy this condition. Let x and y be two regular vertices such that there exists a node a in t such that x and y have the same label at a and are active at a. Let b the lowest node in $\mathbf{tree}(t)$ corresponding to an operation which unifies the labels of x and y. Clearly b corresponds to either a ρ or a 1-$\oplus$-operation. Suppose b corresponds to a 1-$\oplus$-operation. This operation introduces either x or y (say that it introduces x). Since x and y have the same label at b it follows that each neighbor of x is also a neighbor of y. However, since G has minimum degree at least 2, there is a neighbor of x in G'' which is not a neighbor of y, a contradiction.

Let b_1 be the child of b in $\mathbf{tree}(t)$. Clearly x and y are active at b. Since $s_{x,y}$ is the unique vertex in G'' which is adjacent to both x and y, it follows that if we add the edges connecting x and y to $s_{x,y}$ immediately above b_1, then x and y will not be active at b. We show below how to construct an expression t_1 which achieves this goal.

Let t_1' be the expression obtained by removing $s_{x,y}$ from t. Let t_1 be the expression obtained from t_1' by adding immediately above b_1 the vertex $s_{x,y}$ with label $k+2$, then adding two η-operations which connect $s_{x,y}$ to both x and y and then renaming the label of $s_{x,y}$ to $k+1$. (Note that $k+1$ will be a dead label, i.e., no edges will be added to a vertex having label $k+1$.) Since both edges connecting $s_{x,y}$ to x and y already exists at $\mathbf{val}(t_1\langle b\rangle)$, it follows that x and y are not active at $\mathbf{val}(t_1\langle b\rangle)$.

Repeating the above construction for every pair of regular vertices x and y which have the same label at a node a of $\mathbf{tree}(t)$ and are active at a, we finally get a linear $(k+2)$-expression t' which defines G'' and satisfies Property 1.

Note that whenever vertex $s_{x,y}$ gets label $k+2$ at node a of t' it is the unique vertex having this label in $\mathbf{val}(t'\langle a\rangle)$ and thus, it is possible to connect it to x and y using two η-operations. ◻

LEMMA 6. *Let t be a linear k-expression defining G'' that has Property 1. Then there exists a linear $(k+7)$-expression defining G'.*

Proof. Let t be a linear k-expression defining G'' that has Property 1. Let $s = s_{x,y}$ be a special vertex of G''. Thus the neighbors of s in G'' are the regular vertices x and y. Let e_1 and e_2 denote the edges connecting s to x and y, respectively. If the edges e_1 and e_2 are established in t by the same η-operation, then there is a node a in t such that both x and y have the same label at a and are active at a, a contradiction. Thus, we can assume without loss of generality that the edge e_1 is established before e_2 in t. Let a denote the lowest node in $\mathbf{tree}(t)$ corresponding to the η-operation which establishes the edge e_1 in t. Since x and s must have unique labels at a, it follows from Property 1 that node a is the only η-operation in t which connects x to s. Let t_1' denote the expression obtained by removing s from t. Let t_1 denote the expression obtained from t_1' by replacing the node a with the following sequence of operations:

1. Add vertices $s_1, \ldots, s_6$ with labels $k+2, \ldots, k+7$, respectively.
2. Add η-operations connecting s_1, s_2, and s_3 to x.
3. Add η-operations connecting s_1 to s_4, s_2 to s_5, and s_3 to s_6.
4. Add ρ-operations which rename the labels of s_1, s_2, and s_3 to $k+1$ ($k+1$ is used as a dead label).
5. Add ρ-operations which rename the labels of s_4, s_5. and s_6 to ℓ, where ℓ is the label that s has in $\mathbf{val}(t\langle a\rangle)$.

It is easy to check that t_1 defines the graph obtained from G'' by replacing the path of length two $x - s - y$ with the 3 paths of length 3, $x - s_i - s_{i+3} - y$, $i = 1, 2, 3$.

Repeating the above construction for every special vertex s of G'', we finally obtain a linear $(k+7)$-expression t' which defines G'.

Note that whenever vertices $s_1, \ldots, s_6$ get labels $k+2, \ldots, k+7$ at node a of t' they are the unique vertices having these labels in $\mathbf{val}(t'\langle a\rangle)$ and thus, it is possible to establish all the connections and renaming mentioned in steps 2–5 above.

This completes the proof of the lemma. ◻

LEMMA 7. $\text{lin-cwd}(G') \leq \text{lin-cwd}(G'') + 9$.

Proof. Suppose $\text{lin-cwd}(G'') = k$, i.e., there exists a linear k-expression t which defines G''. By Lemma 5 there exists a linear $(k+2)$-expression t_1 which defines G'' and has Property 1. By Lemma 6 there exists a linear $(k+9)$-expression t_2 which defines G'. Thus $\text{lin-cwd}(G') \leq k+9$. ◻

6.2. From G' to G''.

LEMMA 8. $\text{cwd}(G'') \leq \text{cwd}(G') + 3$.

For proving this lemma we shall use the following definitions and lemma.

Let G be a graph and let $\mathcal{D}(G)$ denote the set of graphs which can be obtained from G by replacing each edge of G either with a path of length two or with a path of length three. Clearly, the graph G'' belongs to $\mathcal{D}(G)$ and is obtained by replacing all edges of G with a path of length two. For each graph G^* in $\mathcal{D}(G)$ we call the vertices of G^* which are also vertices of G *regular vertices* and we call the other vertices of G^* *special vertices.*

PROPERTY 2. Let t be a k-expression defining a graph G^* in $\mathcal{D}(G)$. We say that t has *Property 2* if the following conditions hold:

Condition 2.1: There is no η-operation in t which uses label 1, i.e, there is no $\eta_{1,\ell}$-operation in t for any label ℓ. In other words, 1 is a dead label.

Condition 2.2: If label 2 is used in t, then it is used as follows: a special vertex (say s) is introduced with label 2 using a 1-$\oplus$-operation say a, such that s is the only vertex having label 2 at a. Above a in $\mathbf{tree}(t)$ there is a sequence of one or more η-operations followed by a $\rho_{2\to\ell}$-operation where ℓ is any label different from 2 and 3.

Condition 2.3: If label 3 is used in t then it is used as follows: a regular vertex (say r) is introduced with label 3 using a 1-$\oplus$-operation, say a, such that r is the only vertex having label 3 at a. Above a in $\mathbf{tree}(t)$ there is a sequence of operations which can be either η, ρ, or 1-$\oplus$-operations introducing special vertices, followed by a $\rho_{3\to\ell}$-operation where ℓ is any label different from 2 and 3.

Condition 2.4: No regular vertex ever gets label 2 and no special vertex ever gets label 3.

OBSERVATION 1. *Let G^* be a graph in $\mathcal{D}(G)$ and let* $\mathrm{cwd}(G^*) = k$. *Then there is a $(k+3)$-expression t' defining G^* which has Property 2.*

Proof. Let t be a k-expression defining G^*. Let t' be the $k+3$-expression obtained from t by replacing all occurrences of the labels $1, 2$ and 3 with the labels $k+1$,$k+2$ and $k+3$, respectively. Clearly t' defines G^*. Since the labels $1, 2$ and 3 are not used in t', it is obvious that t' has Property 2. □

The following is the key lemma for proving Lemma 8.

LEMMA 9. *Let G^* be a graph in $\mathcal{D}(G)$ and let t be a k-expression which defines G^* and has Property 2. Let a be a lowest node in* $\mathbf{tree}(t)$ *such that there exists an induced path $x - p - q - y$ in G^* (x, y are regular vertices) and x, p, q, y occur at a. Then there exists a k-expression t_1 which has Property 2 and defines the graph G_1^* obtained from G^* by replacing the path $x - p - q - y$ with a path $x - s - y$ where s is a new special vertex.*

Proof. Let a and x, p, q, y as in the statement of the lemma. In each of the following cases we obtain a k-expression t_1 which defines G_1^* and has Property 2. In all cases it is easy to see that the expression t_1 obtained has Property 2.

Case 1: suppose x and y occur on different sides of a. Assume w.l.o.g. that x is on the left side of a and y is on the right side of a.

Case 1.1: suppose that p and q occur on the same side of a. Assume without loss of generality that both p and q occur on the left side of a. Let a_1 denote the lowest node in $\mathbf{tree}(t)$ such that both x and p are in $t\langle a_1\rangle$. Let a_2 denote the lowest node in $\mathbf{tree}(t)$ such that both x and q are in $t\langle a_2\rangle$. By the above assumptions both a_1 and a_2 are descendants of a in $\mathbf{tree}(t)$.

Case 1.1.1: suppose a_1 is a proper descendant of a_2 in $\mathbf{tree}(t)$. If x and q have the same label at a_2 it follows that y must be in $t\langle a_2\rangle$, a contradiction. Thus p and q must have unique labels at a_2. Let ℓ_p and ℓ_q denote the labels of p and q at a_2, respectively.

Case 1.1.1.1: suppose x has a unique label (say ℓ_x) at a_2. In this case, t_1 is obtained from t as follows:

1. Add the following sequence operations immediately above a_2:

1.1. An η_{ℓ_x,ℓ_p}-operation which connects x to p.

1.2. A $\rho_{\ell_p\to\ell_q}$-operation which renames the label of p to the label of q.

2. Omit q.

Case 1.1.1.2: Suppose x does not have unique label at a_2. Thus the edge connecting x to p already exists at $\mathbf{val}(t\langle a_2\rangle)$. In this case, t_1 is obtained from t as follows:

1. Add immediately above a_2 a $\rho_{\ell_p\to\ell_q}$-operation which renames the label of p to the label of q.

2. Omit q.

In both cases 1.1.1.1 and 1.1.1.2, p is connected to y since after p gets the label of q, the η-operation above a which connects q to y will connect p to y. Thus, p can be considered as the new special vertex s in G_1^* and the expression t_1 defines G_1^*.

Case 1.1.2: Suppose a_1 is equal to a_2. In this case x and q must have unique labels at a_2; t_1 is obtained from t as follows:

1. Add immediately above a_2 an η_{ℓ_x,ℓ_q}-operation which connects x and q.

2. Omit p.

In this case it is easy to see that t_1 defines G_1^* and q is the new special vertex s.

Case 1.1.3: suppose a_2 is a proper descendant of a_1 in $\mathbf{tree}(t)$. Since y is not in $t\langle a_1\rangle$, x, p, and q must have unique labels at a_1. Let ℓ_x, ℓ_p, and ℓ_q denote the labels of x, p and q at a_1, respectively. In this case, t_1 is obtained from t as follows:

1. Add the following sequence operations immediately above a_1:

1.1. An η_{ℓ_x,ℓ_p}-operation which connects x to p.

1.2. A $\rho_{\ell_p\to\ell_q}$-operation which renames the label of p to the label of q.

2. Omit q.

As in the previous cases it is easy to see that t_1 defines G_1^* and p is the new special vertex s.

Case 1.2: suppose that p and q occur on different sides of a.

Case 1.2.1: suppose p occurs on the left side of a and q occurs on the right side of a. It is easy to see that at least one of p and q must have a unique label at a. Assume w.l.o.g. that q has a unique label (say ℓ_q) at a. Let ℓ_p and ℓ_y denote the labels that p and y have at a, respectively. Note that y is the only vertex which can have the same label as p at a. In this case, t_1 is obtained from t as follows:

1. Make changes to t such that y will have label ℓ_q at a. In particular let c be the lowest $\oplus$-operation in $\mathbf{tree}(t)$ which contains both y and q. Add a ρ-operation immediately above c which renames the label of y at c to the label of q at c. Note that since q has a unique label ℓ_q at a, we may assume that the label of q at c is also ℓ_q. Then follow the path from c to a in $\mathbf{tree}(t)$ and for each node d corresponding to an η_{ℓ_1,ℓ_2}-operation such that y has label ℓ_1 at d, add an η_{ℓ_q,ℓ_2}-operation immediately above d. Thus, after this step y is connected to all the vertices (except q) which it was connected in $\mathbf{val}(t\langle a\rangle)$ and has label ℓ_q at a.

2. Omit q.

3. After the above changes to y, the label ℓ_p of p at a is unique. Add the following sequence of operations immediately above a:

3.1. An η_{ℓ_p,ℓ_q}-operation which connects y to p.

3.2. A $\rho_{\ell_q\to\ell_y}$-operation which renames y to the label it has in $\mathbf{val}(t\langle a\rangle)$.

By steps 1 and 3.2 above it is clear that all the vertices (except q) which are connected to y in t are also connected to y in t_1. Thus, t_1 defines G_1^* and p is the new special vertex s.

Case 1.2.2: suppose p occurs on the right side of a and q occurs on the left side of a. Since p is adjacent just to x and q, it follows that either x and q have unique labels at a or have the same label at a. If x and q have the same label at a, then there is no way to connect y to q without connecting it also to x, a contradiction. We conclude that the labels at a of p, q, x, and y (say ℓ_p, ℓ_q, ℓ_x and ℓ_y, respectively) are unique. In this case t_1 is obtained from t by omitting q and adding an η_{ℓ_p,ℓ_y}-operation immediately above a.

Case 2: suppose x and y occur on the same side of a. Assume without loss of generality that x and y occur on the left side of a.

Case 2.1: suppose p and q occur on the same side of a. Since a is the lowest node in $\mathbf{tree}(t)$ which contains x, y, p, and q, it follows that p and q must occur on the right side of a. As in case 1.2.2 it is easy to see that the labels at a of p, q, x and y (say ℓ_p, ℓ_q, ℓ_x, and ℓ_y) are unique. In this case t_1 is obtained from t by omitting q and adding an η_{ℓ_p,ℓ_y}-operation immediately above a.

Case 2.2: suppose p and q occur on different sides of a. Assume without loss of generality that p occurs on the left side of a and q occurs on the right side of a. Let a_1 denote the lowest node in $\mathbf{tree}(t)$ which contains both x and p. Let a_2 denote the lowest node in $\mathbf{tree}(t)$ which contains x and y.

Case 2.2.1: suppose a_1 is equal to a_2 or a_2 is a proper descendant of a_1. In this case it is easy to see that x, y and p must have unique labels at a_1 (say ℓ_x, ℓ_y, and ℓ_p, respectively). In this case t_1 is obtained from t by omitting q and adding an η_{ℓ_p,ℓ_y}-operation immediately above a_1.

Case 2.2.2: suppose a_1 is a proper descendant of a_2.

Case 2.2.2.1: suppose y has unique label at a_2 (say ℓ_y). In this case p must have unique label at a_2 (say ℓ_p) and t_1 is obtained from t by omitting q and adding an η_{ℓ_p,ℓ_y}-operation immediately above a_2.

Case 2.2.2.2: suppose y does not have unique label at a_2. Let ℓ_p and ℓ_y denote the labels of p and y at a_2, respectively. Since q is adjacent just to y and p, it follows that p is the only vertex which can share the label of y at a_2. Thus, $\ell_p = \ell_y$. Assume without loss of generality that y is on the right side of a_2 and x and p are on the left side of a_2. Let b_2 denote the right child of a_2 in $\mathbf{tree}(t)$. Note that the complicated handling of this case (as described below) is needed when x is active at a_2 and has the same label as another vertex which is on the right side of a_2. Since q is the only vertex which is adjacent to y and p, it follows that all the vertices which are adjacent to y (except q) must be in $\mathbf{val}(t\langle b_2\rangle)$. Let U denote the set of all vertices (except q) which are adjacent to y. Since y is regular vertex, all vertices in U must be special and have degree exactly 2. For each vertex u in U, let $\mathbf{other}(u)$ denote the neighbor of u which is not y. Let U_1 denote the set of all vertices u in U such that $\mathbf{other}(u)$ is in $\mathbf{val}(t\langle b_2\rangle)$ and let $U_2 = U \setminus U_1$. Let U_{11} denote the set of all vertices u in U_1 such that the lowest node in $\mathbf{tree}(t)$ which contains u and $\mathbf{other}(u)$ does not contain y. Let $U_{12} = U_1 \setminus U_{11}$.

In this case t_1 is obtained from t as follows:

1. Omit q and all vertices of U_2.

2. Let c denote the lowest node in $\mathbf{tree}(t)$ which contains y. Follow the path from c to b_2 in $\mathbf{tree}(t)$ and omit any η_{ℓ_1,ℓ_2}-operation such that the label of y at that point is ℓ_1.

3. Repeat the following step for each u in U_{11}: let c denote the lowest node in $\mathbf{tree}(t)$ which contains u and $\mathbf{other}(u)$. Let d denote the lowest node in $\mathbf{tree}(t)$ which

contains y and u. Since u is in U_{11}, c is a descendant of d. Thus, u and $\mathbf{other}(u)$ have unique labels at c (say ℓ_u and ℓ, respectively). Add an $\eta_{\ell_u,\ell}$-operation immediately above c which connects u and $\mathbf{other}(u)$. Add a ρ-operation immediately above d which renames the label of u to the label of y at d. Thus, after step 3 each vertex u in U_{11} is connected to $\mathbf{other}(u)$ and has label ℓ_y at a_2.

4. Repeat the following step for each u in U_{12}: let c denote the lowest node in $\mathbf{tree}(t)$ which contains u and $\mathbf{other}(u)$.

4.1. Suppose $\mathbf{other}(u)$ is a special vertex. If $\mathbf{other}(u)$ does not have a unique label at c then its label at c must be equal to the label of y at c, a contradiction, since q distinguishes y and $\mathbf{other}(u)$. Thus, $\mathbf{other}(u)$ must have unique label at c. If u does not have unique label at c, then the label of u at c must be equal to the label of the unique regular vertex (say z) which is adjacent to $\mathbf{other}(u)$. But then vertices of the induced path $z - \mathbf{other}(u) - u - y$ of G^* occur at a_2, and since a_2 is a descendant of a, we have a contradiction to the selection of a as a lowest such node with that property. We conclude that both u and $\mathbf{other}(u)$ have unique labels at c. Thus, in this case add an η-operation immediately above c connecting u and $\mathbf{other}(u)$ and above it add a ρ-operation which renames the label of u to the label that y has at that point.

4.2. Suppose $\mathbf{other}(u)$ is a regular vertex. Since t has Property 2, it follows that either label 2 is not used at c or u is the only vertex having label 2 at c. In this case omit u from t and add the following sequence of operations immediately above c:

4.2.1. A 1-$\oplus$-operation introducing u with label 2.

4.2.2. An $\eta_{2,\ell}$-operation, where ℓ is the unique label that $\mathbf{other}(u)$ has at c.

4.2.3. A $\rho_{2\to\ell'}$-operation where ℓ' is the unique label that y has at c.

Thus, after step 4 each vertex u in U_{12} is connected to $\mathbf{other}(u)$ and has label ℓ_y at a_2.

5. Omit y from t and add the following sequence of operations immediately above a_2:

5.1. A 1-$\oplus$-operation which introduces y with label 3. Note that since t has Property 2 label 3 is not used at a_2.

5.2. An η_{3,ℓ_y}-operation connecting y to p and all the vertices in U_1.

5.3. A $\rho_{\ell_y\to 1}$-operation renaming p and all the vertices in U_1 to a dead label.

5.4. For each vertex u in U_2 add the following sequence of operations:

5.4.1. A 1-$\oplus$-operation introducing u with label 2.

5.4.2. An $\eta_{2,3}$-operation connecting u and y.

5.4.3. A $\rho_{2\to\ell}$-operation where ℓ is the label that u has in t at a_2.

Thus after step 5.4 all the vertices in U_2 are connected to y and have the same label as they have in t at a_2.

5.5. A $\rho_{3\to 1}$-operation renaming the label of y to a dead label.

Each vertex u in U_1 is connected to $\mathbf{other}(u)$ in step 3 or in step 4 and is connected to y in step 5.2. Each vertex u in U_2 is connected to y at step 5.4.2 and the η-operation in t above a_2 which connects u to $\mathbf{other}(u)$ also exists in t_1 and connects u to $\mathbf{other}(u)$ since after step 5.4 the label of u is the same as its label at a_2 in t.

Thus, t_1 defines G_1^* and p is the new special vertex s.

This completes the proof of Lemma 9. ∎

Proof of Lemma 8. Suppose $\mathrm{cwd}(G') = k$. Let G_1' denote the induced subgraph of G' obtained by removing from G' for every edge $e = xy$ of G, the two pairs of vertices p_i, q_i, $i = 1, 2$, where $x - p_i - q_i - y$ are two of the three paths of length 3 between x and y. Since G_1' is an induced subgraph of G', it follows that $\mathrm{cwd}(G_1') \leq k$. Clearly,

G_1' belongs to $\mathcal{D}(G)$. Let t be a k-expression which defines G_1'. By Observation 1, there is a $(k+3)$-expression t' defining G_1' which has Property 2. Let a be a lowest node in $\mathbf{tree}(t')$ such that for an induced path $x-p-q-y$ of G_1' (x and y are regular vertices) the vertices x, p, q, y occur at a. By Lemma 9 there exists a $(k+3)$-expression t_1' which has Property 2 and defines the graph G_1^* obtained from G_1' by replacing the path $x-p-q-y$ with a path $x-s-y$ where s is a new special vertex. We can repeat this process until we finally get a $(k+3)$-expression t'' which defines the graph G'' that is obtained from G_1' by replacing all induced paths of length 3 (with regular end vertices and special internal vertices) by induced paths of length 2. This completes the proof of Lemma 8. $\square$

7. Comparing clique-width and linear clique-width.

As discussed in Section 2.3, the difference between the clique-width and the linar clique-width of a graph can be arbitrarily large. In this section we show that this is not true for graphs obtained by means of Construction 3 from a cobipartite graphs of minimum degree at least 2.

THEOREM 5. *If G is a cobipartite graph with minimum degree at least 2, then* lin-cwd$(G'') \leq$ cwd$(G'') + 6$.

In this section we assume that G is a cobipartite graph with minimum degree at least 2. Since G is cobipartite the vertices of G can be partitioned into two cliques A and B. The regular vertices of G'' which belong to A, B are called *A-regular* vertices, *B-regular vertices*, respectively.

Let t be a cwd-expression defining G''. Let a be a $\oplus$-operation of t. We say that there is a *separation* at a between the A-regular vertices and the B-regular vertices if all A-regular vertices of $\mathbf{val}(t\langle a\rangle)$ occur on one side of a (say, on the left side of a) and all the B-regular vertices of $\mathbf{val}(t\langle a\rangle)$ occur on the other side of a (say, on the right side of a).

For the proof of Theorem 5, we start with a k-expression t defining G'' and show with a series of lemmas (Lemmas 10–15) how to construct a $(k+6)$-expression t' which defines G'' and includes at most one (>1)-$\oplus$-operation, say a, such that in a there is a separation between the A-regular and the B-regular vertices. Finally, we show with Lemma 16 how to construct from t' a *linear* $(k+6)$-expression defining G''; this is done by omitting the vertices on the left side of a and by introducing them instead on the right side of a, one after the other using 1-$\oplus$-operations.

PROPOSITION 1. *Let t be a* cwd-*expression defining G''. For each $\oplus$-operation a of t there is at most one pair of A-regular (B-regular) vertices which occur on different sides of a and have the same label at a.*

Proof. Suppose there are two different pairs $\{x_1, y_1\}$ and $\{x_2, y_2\}$ of A-regular vertices such that for $i = 1, 2$, x_i and y_i occur at different sides of a and have the same label at a. Assume without loss of generality that x_1 and x_2 occur on the left side of a and y_1 and y_2 occur on the right side of a. Clearly, either $x_1 \neq x_2$ or $y_1 \neq y_2$. Assume without loss of generality that $x_1 \neq x_2$. Consider the special vertex s_{y_1,x_2}. If s_{y_1,x_2} is not in $\mathbf{val}(t\langle a\rangle)$, then when later on the edge connecting s_{y_1,x_2} to y_1 will be establish, also the edge connecting it to x_1 will be established, a contradiction. Thus s_{y_1,x_2} is in $\mathbf{val}(t\langle a\rangle)$. If s_{y_1,x_2} occurs on the left side of a then when the edge connecting it to y_1 will be established, it will be connected also to x_1, a contradiction. If s_{y_1,x_2} is on the right side of a, then when the edge connecting it to x_2 will be established, it will be connected also to y_2. Since the degree of s_{y_1,x_2} in G'' is exactly 2, it follows that

y_1 must be equal to y_2. Thus, the three vertices x_1, x_2 and y_1 have the same label at a, which implies that the η-operation above a which connect s_{y_1,x_2} to x_2 connect it also to x_1, a contradiction. The argument for two different pairs of B-regular vertices is symmetric. ◻

PROPOSITION 2. *Let t be a* cwd-*expression defining G''. Let a be a $\oplus$-operation of t and let $\{x_1, y_1\}$ be a pair of A-regular (B-regular) vertices which occur on different sides of a and have the same label at a. Then at least one of x_1 and y_1 is active at a and for every other vertex (say z) occurring at a the label of z is different from the label of x_1 and y_1 at a.*

Proof. Suppose that both x_1 and y_1 are dead at a. Since x_1 is dead at a the special vertex s_{x_1,y_1} must be on the same side of a as x_1. Similarly, s_{x_1,y_1} must be on the same side of a as y_1, a contradiction, since x_1 and y_1 occur on different sides of a. If there is another vertex z with the same label as x_1 and y_1 at a, then, when the edges connecting some vertex of G'' (say, w) to x_1 or y_1 will be established (such edges must be established since either x_1 or y_1 is active at a), also the edge connecting it to z will be established, a contradiction (no vertex of G'' is adjacent to x_1, y_1 and z). ◻

PROPOSITION 3. *Let t be a* cwd-*expression defining G''. Let a be an $\oplus$-operation of t and let $\{x_1, y_1\}$ be a pair of regular vertices which occur on different sides of a and have the same label at a. Then all the edges connecting x_1 (y_1) to its neighbors in $G'' - s_{x_1,y_1}$ exist in $\mathbf{val}(t\langle a\rangle)$.*

Proof. Let s be a vertex which is adjacent to x_1 in $G'' - s_{x_1,y_1}$. Clearly s must be a special vertex of the form $s_{x_1,z}$ for $z \neq y_1$. If s is not connected to x_1 in $\mathbf{val}(t\langle a\rangle)$, then it is not possible to connect s to x_1 without connecting it also to y_1, a contradiction. ◻

7.1. Property 3.

PROPERTY 3. We say that t has *Property 3* if the following conditions hold for t:

Condition 3.1: The label 1 is dead in t.

Condition 3.2: For each (> 1)-$\oplus$-operation a in t, there is no pair of A-regular (B-regular) vertices which occur on different sides of a and have the same label at a.

LEMMA 10. *Let t be a k-expression defining G''. Then there exists a $(k + 4)$-expression t' defining G'' such that t' has Property 3.*

Proof. Let t be a k-expression defining G''. Let t_1 denote the $(k + 1)$-expression obtained from t by replacing each occurrence of the label 1 with the label $k + 1$. Clearly, t_1 defines G'' and label 1 is dead in t_1. Let a be a (> 1)-$\oplus$-operation in t_1 such that there exist at least one pair of regular vertices that violate Condition 3.2. We define below a $(k + 4)$-expression t_2 which defines G'' and has the additional property that there is no pair of regular vertices of the same type which occur on different sides of a and have the same label in $\mathbf{val}(t_2\langle a\rangle)$. Let b denote the left child of a in $\mathbf{tree}(t)$.

By Proposition 1, one of the following cases must prevail.

Case 1: Suppose there is exactly one pair (say $\{x_1, y_1\}$) of regular vertices of the same type which occur on different sides of a and have the same label in $\mathbf{val}(t_1\langle a\rangle)$. Assume without loss of generality that x_1 occurs on the left side of a. By Proposition 2, either x_1 or y_1 must be active at a and their label at a (say ℓ) is different from the labels of all the other vertices at a. In this case t_2 is obtained from t_1 as follows:

1. Add a $\rho_{\ell\to k+2}$-operation immediately above b.
2. Omit s_{x_1,y_1}.
3. Add the following sequence of operations immediately above a:

3.1. A 1-$\oplus$-operation introducing s_{x_1,y_1} with label $k + 4$.

3.2. An $\eta_{k+4,\ell}$-operation which connects s_{x_1,y_1} to y_1.

3.3. An $\eta_{k+4,k+2}$-operation which connects s_{x_1,y_1} to x_1.

3.4 A $\rho_{k+4\to 1}$-operation renaming the label of s_{x_1,y_1} to a dead label.

3.5 A $\rho_{k+2\to 1}$-operation renaming the label of x_1 to a dead label.

3.6 A $\rho_{\ell\to 1}$-operation renaming the label of y_1 to a dead label.

Case 2: Suppose there are exactly two pairs (say $\{x_1, y_1\}$ and $\{x_2, y_2\}$) of regular vertices of the same type which occur on different sides of a and have the same label in $\mathbf{val}(t_1\langle a\rangle)$. Assume without loss of generality that x_1 and x_2 occur on the left side of a. By Proposition 2, either x_1 or y_1 is active at a and their label at a (say ℓ_1) is different from the labels of all the other vertices at a. Similarly, x_2 and y_2 have the same unique label at a (say ℓ_2). It follows that all the vertices x_1, x_2, y_1, y_2 are distinct.

In this case t_2 is obtained from t_1 as follows:

1. Add the following sequence of operations immediately above b:

1.1 A $\rho_{\ell_1\to k+2}$-operation renaming the label of x_1 to to $k+2$.

1.1 A $\rho_{\ell_2\to k+3}$-operation renaming the label of x_2 to to $k+3$.

2. Omit s_{x_1,y_1} and s_{x_2,y_2}.

3. Add the following sequence of operations immediately above a:

3.1. A 1-$\oplus$-operation introducing s_{x_1,y_1} with label $k+4$.

3.2. An η_{k+4,ℓ_1}-operation which connects s_{x_1,y_1} to y_1.

3.3. An $\eta_{k+4,k+2}$-operation which connects s_{x_1,y_1} to x_1.

3.4 A $\rho_{k+4\to 1}$-operation renaming the label of s_{x_1,y_1} to a dead label.

3.5. A 1-$\oplus$-operation introducing s_{x_2,y_2} with label $k+4$.

3.6. An η_{k+4,ℓ_2}-operation which connects s_{x_2,y_2} to y_2.

3.7. An $\eta_{k+4,k+3}$-operation which connects s_{x_2,y_2} to x_2.

3.8 A sequence of ρ-operations renaming all labels $\ell_1, \ell_2, k+2, k+3, k+4$, to the dead label 1.

In both cases 1 and 2 it follows from Proposition 3 that the expression t_2 defines G''.

Repeating the above procedure for every (>1)-$\oplus$-operation in t_2 we finally get a $(k+4)$-expression t' defining G'' such that t' has Property 3. ▯

7.2. Property 4.

The following property is similar to Property 2.

Property 4. Let t be a k-expression defining G'' which has Property 3. We say that t has *Property 4*, if the following conditions hold:

Condition 4.1: If label 2 is used in t, then it is used as follows: a special vertex (say s) is introduced with label 2 using a 1-$\oplus$-operation say a, such that s is the only vertex having label 2 at a. Above a in $\mathbf{tree}(t)$ there is a sequence of one or more η-operations followed by a $\rho_{2\to\ell}$-operation where ℓ is any label different from 2 and 3.

Condition 4.2: If label 3 is used in t then it is used as follows: a regular vertex (say r) is introduced with label 3 using a 1-$\oplus$-operation, say a, such that r is the only vertex having label 3 at a. Above a in $\mathbf{tree}(t)$ there is a sequence of operations which can be either η, ρ, or 1-$\oplus$-operations introducing special vertices, followed by a $\rho_{3\to\ell}$-operation where ℓ is any label different from 2 and 3.

Condition 4.3: No regular vertex ever gets label 2 and no special vertex ever gets label 3.

Lemma 11. *Let t be a k-expression defining G'' such that t has Property 3. Then there exists a $(k+2)$-expression t' defining G'' such that t' has Property 4.*

Proof. Let t be a k-expression defining G'' such that t has Property 3. Let t' denote the $(k+2)$-expression obtained from t by replacing each occurrence of the label 2 with the label $k+1$ and replacing each occurrence of the label 3 with the label

$k+2$. Clearly, t' defines G''. Since labels 2 and 3 are not used in t', it is obvious that t' has Property 4. □

7.3. Property 5.

PROPERTY 5. Let t be a k-expression defining G'' which has Property 4. We say that t has *Property 5*, if the following condition holds:

Condition 5: For each (>1)-$\oplus$-operation a in t, there is no regular vertex which occurs at a and has a unique label at a which is different from label 1.

LEMMA 12. *Let t be a k-expression defining G'' such that t has Property 4. Then there exists a k-expression t' defining G'' such that t' has Property 5.*

For proving this lemma we use the following definitions and lemma. Let t be a k-expression defining G''. For each (>1)-$\oplus$-operation a in t let $n(t\langle a\rangle)$ denote the number of regular vertices which occur at a and have unique labels at a which are different from label 1. Let $n(t)$ denote the sum of $n(t\langle a\rangle)$ over all (>1)-$\oplus$-operations in t. Clearly, if a k-expression t defines G'' and has Property 4, then $n(t)=0$ implies that t has also Property 5.

LEMMA 13. *Let t be a k-expression defining G'' such that t has Property 4 and $n(t)>0$. Then there exists a k-expression t' defining G'' such that t' has Property 4 and $n(t')<n(t)$.*

Proof. Let t be a k-expression defining G'' such that t has Property 4 and $n(t)>0$. Since $n(t)>0$, there exists a (>1)-$\oplus$-operation a in t and a regular vertex x such that x has unique label (say ℓ_x) in $\mathbf{val}(t\langle a\rangle)$. We will construct below a k-expression t' defining G'', such that in t', x is introduced by a 1-$\oplus$-operation above a. We shall use the following notation and proceed similarly as in the proof of Lemma 9. Let b denote the child of a in $\mathbf{tree}(t)$ such that x is in $\mathbf{val}(t\langle b\rangle)$. Let U denote the set of all vertices which are adjacent to x and occur in $\mathbf{val}(t\langle b\rangle)$. Since x is a regular vertex, all vertices in U must be special and have degree exactly 2. For each vertex $u\in U$, let $\mathbf{other}(u)$ denote the neighbor of u which is not x. Let U_1 denote the set of all vertices $u\in U$ such that $\mathbf{other}(u)$ is in $\mathbf{val}(t\langle b\rangle)$ and let $U_2=U\setminus U_1$. Let U_{11} denote the set of all vertices $u\in U_1$ such that the lowest node in $\mathbf{tree}(t)$ which contains u and $\mathbf{other}(u)$ does not contain x. Let $U_{12}=U_1\setminus U_{11}$. The k-expression t' is obtained from t as follows:

1. Omit all vertices of U_2.

2. Let c denote the lowest node in $\mathbf{tree}(t)$ which contains x. Follow the path from c to b in $\mathbf{tree}(t)$ and omit any η_{ℓ_1,ℓ_2}-operation such that the label of x at that point is ℓ_1.

3. Repeat the following step for each $u\in U_{11}$: let d denote the lowest node in $\mathbf{tree}(t)$ which contains u and $\mathbf{other}(u)$. Let e denote the lowest node in $\mathbf{tree}(t)$ which contains x and u. Since u is in U_{11}, d is a descendant of e. Thus, u and $\mathbf{other}(u)$ have unique labels at d (say ℓ_u and ℓ, respectively). Add an $\eta_{\ell_u,\ell}$-operation immediately above d which connects u and $\mathbf{other}(u)$. Add a ρ-operation immediately above e which renames the label of u to the label of x at e. Thus, after step 3 each vertex $u\in U_{11}$ is connected to $\mathbf{other}(u)$ and has label ℓ_x at a.

4. Repeat the following step for each $u\in U_{12}$: let d denote the lowest node in $\mathbf{tree}(t)$ which contains u and $\mathbf{other}(u)$. Since t has Property 4, and u and $\mathbf{other}(u)$ occur on different sides of d it follows that the only vertex which can have label 2 at d is u. Omit u from t and add the following sequence of operations immediately above d:

4.1. A 1-$\oplus$-operation introducing u with label 2.

4.2. An $\eta_{2,\ell}$-operation connecting u and $\mathbf{other}(u)$, where ℓ is the unique label that $\mathbf{other}(u)$ has at d.

4.2.3. A $\rho_{2\to\ell'}$-operation where ℓ' is the unique label that x has at d.

Thus, after step 4 each vertex $u \in U_{12}$ is connected to $\mathbf{other}(u)$ and has label ℓ_x at a.

5. Omit x from t and add the following sequence of operations immediately above a:

5.1. A 1-$\oplus$-operation which introduces x with label 3. Note that since t has Property 4 and a is a (> 1)-$\oplus$-operation label 3 is not used at a.

5.2. An η_{3,ℓ_x}-operation connecting x to all the vertices in U_1.

5.3. A $\rho_{\ell_x\to 1}$-operation renaming the label of all the vertices in U_1 to a dead label.

5.4. For each vertex $u \in U_2$ add the following sequence of operations:

5.4.1. a 1-$\oplus$-operation introducing u with label 2;

5.4.2. an $\eta_{2,3}$-operation connecting u to x;

5.4.3. a $\rho_{2\to\ell}$-operation where ℓ is the label that u has in t at a.

Thus after step 5.4 all the vertices in U_2 are connected to x and have the same label as they have in t at a.

5.5. A $\rho_{3\to\ell_x}$-operation renaming the label of x to the label it has in $\mathbf{val}(t\langle a\rangle)$.

Each vertex $u \in U_1$ is connected to $\mathbf{other}(u)$ in step 3 or in step 4 and is connected to x in step 5.2. Each vertex $u \in U_2$ is connected to x at step 5.4.2 and the η-operation in t above a which connects u to $\mathbf{other}(u)$ also exists in t' and connects u to $\mathbf{other}(u)$. Since after step 5.5. the label of x is the same as its label in $\mathbf{val}(t\langle a\rangle)$, it follows that all the vertices which are adjacent to x and are not in U will be connected to x in t' by the same η-operations which connect them to x in t.

Thus, t' defines G''. Since the above changes to t did not violate the rules of Property 4, it follows that t' has Property 4. Finally, since in t', x is introduced by a 1-$\oplus$-operation above a, and all other regular vertices are not moved, it follows that $n(t') < n(t)$. This completes the proof of Lemma 13. ☐

Proof of Lemma 12. The result follows easily by applying Lemma 13 (at most) $n(t)$ times until a k-expression t' is obtained such that t' defines G'' and $n(t') = 0$. ☐

PROPOSITION 4. *Let t be a k-expression defining G'' such that t has Property 5. Let a be a (> 1)-$\oplus$-operation in t such that at least one regular vertex occurs on the left side of a and at least one regular vertex occurs on the right side of a. Then there is a separation at a between the A-regular and the B-regular vertices.*

Proof. Let a be a (> 1)-$\oplus$-operation in t and let x and y be two regular vertices occurring on different sides of a. Assume without loss of generality that x occurs on the left side of a and y occurs on the right side of a. Suppose x and y are both A-regular vertices. By Condition 3.2, x and y do not have the same label at a. Suppose x or y (say x) has label 1 at a. By Condition 5, there exists vertex z which have the same label as y at a. The special vertex $s = s_{x,y}$ must occur on the left side of a, or else no η-operation connect s and x in t, a contradiction. Thus, the η-operation above a in $\mathbf{tree}(t)$ which connects s to y connects it also to z, a contradiction. We conclude that both x and y do not have label 1 at a. By Condition 5, there are two vertices w and z which have the same label as x and y at a, respectively. Let $s = s_{x,y}$. If s does not occur at a, then the η-operation in t which connects s to x, connects it also to w, a contradiction. If s occurs on the left side of a, then the η-operation which connects s to y connects it also to z, a contradiction. If s occurs on the right side of a, then the η-operation which connects s to x connects it also to w, a contradiction. Thus x

and y can not be both A-regular vertices.

Similarly, x and y cannot be both B-regular vertices. Thus, one of x and y (say, x) must be A-regular and the other (say, y) must be B-regular. If there is a B-regular vertex (say, z) on the left side then there are two B-regular vertices (z and y) occurring on different sides of a, which is not possible by the above argument. Thus all the A-regular vertices occur on the left side of a and all the B-regular vertices occur on the right side of a. ☐

7.4. Property 6.

PROPERTY 6. Let t be a k-expression defining G''. We say that t has *Property 6* if it has Property 5 and the following condition holds:

Condition 6: Either there are no (>1)-$\oplus$-operations in t or there is just one (>1)-$\oplus$-operation in t (say, a) and there is a separation at a between the A-regular and the B-regular vertices.

LEMMA 14. *Let t be a k-expression defining G'' such that t has Property 5. Then there exists a k-expression t' which defines G'' and has Property 6.*

Proof. Let t be a k-expression which defines G'' and has Property 5. Let a be a (>1)-$\oplus$-operation in t such that one side of a (say, the left side) contains just special vertices (say, $s_1,\dots,s_m$). Clearly, $s_1,\dots,s_m$ are isolated vertices in $\mathbf{val}(t\langle a\rangle)$ and have unique labels in $\mathbf{val}(t\langle a\rangle)$. Let $\ell_1,\dots,\ell_m$ denote the labels of $s_1,\dots,s_m$ in $\mathbf{val}(t\langle a\rangle)$, respectively. Let b be the right child of a. Let t_1 be the expression obtained from t by replacing $t\langle a\rangle$ with

$$t\langle b\rangle \oplus \ell_1(s_1) \oplus \ldots \oplus \ell_m(s_m).$$

It is easy to verify that t_1 also defines G'' and has Property 5.

Let t' denote the expression obtained from t_1 by repeating the above process for each (>1)-$\oplus$-operation a in t_1 such that one side of a contains just special vertices. Let a be a (>1)-$\oplus$-operation in t'. By the above construction, each side of a contains at least one regular vertex. By Proposition 4, since Property 5 holds for t', there is a separation at a in t' between the A-regular vertices and the B-regular vertices. Suppose there is another (>1)-$\oplus$-operation (say a') in t'. By the above argument each side of a' contains at least one regular vertex and there is a separation at a' in t' between the A-regular and the B-regular vertices. If a is a descendant of a' in $\mathbf{tree}(t')$, then there cannot be a separation at a' between the A-regular and the B-regular vertices, a contradiction. Similarly, a' is not a descendant of a in $\mathbf{tree}(t')$. Let a'' be the lowest node in $\mathbf{tree}(t')$ which contains both a and a'. Clearly a'' must be a (>1)-$\oplus$-operation. By Proposition 4 there is a separation at a'' in t' between the A-regular and the B-regular vertices. Since a occurs on one side of a'', this side of a'' contains both A-regular and B-regular vertices, a contradiction. We conclude that a is a unique (>1)-$\oplus$-operation in t'. Thus t' is a k-expression which defines G'' and has Property 6. ☐

7.5. Property 7.

PROPERTY 7. Let t be a k-expression defining G''. We say that t has *Property 7* if it has *Property 6* and either t is linear or the following condition holds:

Condition 7: Let a be the unique (>1)-$\oplus$-operation in t. Then for each A-regular (B-regular) vertex x, which is active at a and occurs on one side (say left side) of a, there is a unique B-regular (A-regular) vertex y which is active at a and occurs on the other side (say right side) of a and has the same label as x in $\mathbf{val}(t\langle a\rangle)$.

LEMMA 15. *Let t be a k-expression defining G'' such that t has Property 6. Then there exists a k-expression t' which defines G'' and has Property 7.*

Proof. Let a be the unique (>1)-$\oplus$-operation in t. Assume without loss of generality that all the A-regular vertices of $\mathbf{val}(t\langle a\rangle)$ occur on the left side of a and all the B-regular vertices of $\mathbf{val}(t\langle a\rangle)$ occur on the right side of a. Let x be a regular vertex which is active at a. Let ℓ denote the label of x at a. Since Condition 5 holds for t, the label of x at a is not unique. Suppose there are two vertices u and v which are distinct from x and have label ℓ at a. Since x is active at a, there is an η-operation above a in $\mathbf{tree}(t)$ which connects some special vertex (say, s) to x. This η-operation connects s also to u and v, a contradiction (since s is adjacent in G'' to exactly two vertices). Thus, for each regular vertex x which has label ℓ at a and is active at a there is a unique second vertex (say y) which is active at a and has label ℓ at a. By a similar argument no η-operation above a in $\mathbf{tree}(t)$ connects a vertex other than $s_{x,y}$ to x or to y. Thus, all edges incident to x or y in G'', except $xs_{x,y}$ and $ys_{x,y}$, already exist in $\mathbf{val}(t\langle a\rangle)$.

We now define the cwd-expression t_1 depending on the following cases:

Case 1: One of the vertices x, y is A-regular and one is B-regular. Since Condition 7 holds in this case for x and y we set $t_1 = t$.

Case 2: Both x and y are A-regular. Let b denote the left child of a. In this case t_1 is obtained from t as follows:

1. Omit $s_{x,y}$ from t.

2. Add immediately above b the following sequence of operations:

2.1. A 1-$\oplus$-operation which introduces $s_{x,y}$ with label 2. Note that since t has Property 4, and a is a (>1)-$\oplus$-operation, label 2 is not used in $\mathbf{val}(t\langle a\rangle)$.

2.2. An $\eta_{2,\ell}$-operation which connects $s_{x,y}$ to x and y, where ℓ is the label that x and y have in $\mathbf{val}(t\langle b\rangle)$.

2.3. A $\rho_{2\to1}$-operation renaming the label of $s_{x,y}$ to the dead label 1.

2.4. A $\rho_{\ell\to1}$-operation renaming the label of x and y to the dead label 1.

Case 3: Both x and y are B-regular. This case is symmetric to Case 2.

Let t' denote the expression obtained by repeating the above process for each regular vertex which is active at a. It is easy to see that t' defines G'' and has Property 7, as required. ▯

7.6. Linear expressions for G''.

Lemma 16. *Let t be a k-expression defining G'' such that t has Property 7. Then there is a linear k-expression which defines G''.*

In the proof of Lemma 16 we shall use the following definition and proposition.

Let t be a cwd-expression which defines G'', let a be any node of $\mathbf{tree}(t)$ and let $s_{x,y}$ be any special vertex in $\mathbf{val}(t\langle a\rangle)$. The label of $s_{x,y}$ at a is called an *x-connecting label* at a (a *y-connecting label* at a) if $\mathbf{val}(t\langle a\rangle)$ includes the edge connecting $s_{x,y}$ to y (x) but does not include the edge connecting $s_{x,y}$ to x (y).

Proposition 5. *Let t be a* cwd*-expression which defines G'', let a be any node of* $\mathbf{tree}(t)$*, and let y_1, y_2 be two distinct regular vertices of G''. Suppose that there is a y_1-connecting label and a y_2-connecting label at a. Then these two labels are different.*

Proof. Let s_1 and s_2 be two special vertices that have a y_1-connecting label and a y_2-connecting label at a, respectively. By definition, s_1 is a special vertex of the form s_{x_1,y_1} where s_1 is connected to x_1 and is not connected to y_1 in $\mathbf{val}(t\langle a\rangle)$. Similarly, s_2 is a special vertex of the form s_{x_2,y_2} where s_2 is connected to x_2 and is not connected to y_2 in $\mathbf{val}(t\langle a\rangle)$. Suppose that the labels of s_1 and s_2 are the same in $\mathbf{val}(t\langle a\rangle)$. The η-operation above a which connects s_1 to y_1 connects also s_2 to y_1. Thus s_2 is connected to x_2, y_2 and y_1. Since $y_1 \neq y_2$ and $x_2 \neq y_2$ and s_2 has degree 2, it follows that $x_2 = y_1$. By a symmetric argument we get that x_1 is equal to y_2. We conclude that $s_1 = s_2$. But this is not possible since $s_1 = s_2$ is connected to x_1 and is not connected to $y_2 = x_1$. ◻

7.7. Proof of Lemma 16. If there is no (> 1)-$\oplus$-operation in t, the claim follows immediately. Let a be the unique (> 1)-$\oplus$-operation in t. Let b and c denote the left child and the right child of a in $\mathbf{tree}(t)$, respectively. Assume without loss of generality that all the regular vertices in $\mathbf{val}(t\langle b\rangle)$ are A-regular and all regular vertices in $\mathbf{val}(t\langle c\rangle)$ are B-regular.

First we introduce the following notation. Let A_1 (B_1) denote the set of A-regular (B-regular) vertices of $\mathbf{val}(t\langle b\rangle)$ ($\mathbf{val}(t\langle c\rangle)$) and put $A_2 = A \setminus A_1$ and $B_2 = B \setminus B_1$. Let $\mathbf{Active}(A_1)$ ($\mathbf{Active}(B_1)$) denote the set of vertices of A_1 (B_1) which are active at a. Let $\mathbf{Dead}(A_1)$ ($\mathbf{Dead}(B_1)$) denote the set of vertices of A_1 (B_1) which are dead at a. Clearly, $A_1 = \mathbf{Active}(A_1) \cup \mathbf{Dead}(A_1)$ and $B_1 = \mathbf{Active}(B_1) \cup \mathbf{Dead}(B_1)$. By Condition 7, $|\mathbf{Active}(A_1)| = |\mathbf{Active}(B_1)|$. For each B-regular vertex $u \in \mathbf{Active}(B_1)$ we denote by $\mathbf{mate}(u)$ the unique A-regular vertex (guaranteed by Condition 7) which is in $\mathbf{Active}(A_1)$ and has the same label as u in $\mathbf{val}(t\langle a\rangle)$. Let $|\mathbf{Dead}(A_1)| = q$. Let x_i, $1 \le i \le q$, be the ith vertex in $\mathbf{Dead}(A_1)$ which gets a non-unique label or label 1 in $t\langle b\rangle$ (if there is more than one such vertex, choose one of them arbitrarily) and let w_i be the highest node in $\mathbf{tree}(t\langle b\rangle)$ such that x_i has a unique label (which is different from label 1) in $t\langle w_i\rangle$. Note that w_i is well defined since each regular vertex in G'' is a leaf of $\mathbf{tree}(t)$ having a unique initial label (which is different from label 1).

Let $X_i = \{x_1, \ldots, x_i\}$, $1 \le i \le q$. Let NX_i, $1 \le i \le q$, denote the set of B-regular vertices which have a neighbor (in G) in the set X_i. For convenience we set $NX_0 = \emptyset$.

For the proof of Lemma 16, we start with a k-expression t defining G'' which has a unique (> 1)-$\oplus$-operation a, such that there is a separation at a between the A-regular and the B-regular vertices, and show how to construct a linear k-expression t' which defines G'', by omitting the vertices on the left side of a and introducing them using 1-$\oplus$-operations one after the other on the right side of a. In particular, at the top of the right side of a the following regular vertices are introduced in this order: the vertices of $\mathbf{Active}(A_1)$ (see the expression e_2 below), the vertices of $A_2 \cup B_2$ (see the expression e_3 below), and the vertices of $\mathbf{Dead}(A_1)$ in the order $x_q, .., x_1$ (see the expressions f_i below). We show in a sequence of observations that for each regular vertex x all its neighbors in G'' exist in the linear cwd-expression t' that we construct. Finally, we use the bounds on k, obtained in Observations 4, 5, and 7, to show that the number of labels used in t' is at most k.

Observation 2. *Let v be a vertex which is adjacent to x_i (in G) and is not in $\mathbf{val}(t\langle w_i\rangle)$. Then the special vertex $s_{x_i,v}$ has the v-connecting label at w_i.*

Proof. Suppose the vertex $s = s_{x_i,v}$ is not adjacent to x_i in $\mathbf{val}(t\langle w_i\rangle)$. Let w_i' denote the parent of w_i in $\mathbf{tree}(t)$. The label of x_i at w_i' is either 1 or the label of another vertex (say u)). If the label of x_i at w_i' is 1 then no η-operation in t connects s and x_i, a contradiction. Thus, the label of x_i is is the same as the label of u at w_i'. If $u \ne v$ then the η-operation above w_i' which connects s to x_i connects it also to u, a contradiction. If $u = v$ then w_i' must correspond to a 1-$\oplus$-operation which introduces v with the label of x_i. Since v and x_i have the same label at w_i' it follows that each neighbor of v is also a neighbor of x_i. However, since G has minimum degree at least 2, there is a neighbor of v in G'' which is not a neighbor of x_i, a contradiction. ◻

Observation 3. *For $1 \le i \le q$, $\mathbf{labels}(\mathbf{val}(t\langle w_i\rangle)) \ge |A| + |NX_i| + 1 - i$.*

Proof. Let v be a vertex in $\mathbf{Active}(A_1)$. If v occurs at w_i, then v has a unique label at $\mathbf{val}(t\langle w_i\rangle)$. If v does not occur at w_i, then by Observation 2 the vertex $s_{x_i,v}$ has a v-connecting label at w_i. Thus, so far we have $|\mathbf{Active}(A_1)|$ different labels in $\mathbf{val}(t\langle w_i\rangle)$. Let v be a vertex in $\mathbf{Dead}(A_1)\setminus X_i$. If v occurs at w_i, then by definition v must have a unique label at w_i. If v does not occur at w_i, then by Observation 2 the vertex $s_{x_i,v}$ has a v-connecting label at w_i. Thus, by Proposition 5, we have additional $|\mathbf{Dead}(A_1)\setminus X_i| = q-i$ labels in $\mathbf{val}(t\langle w_i\rangle)$. Let v be a vertex in A_2. By Observation 2, the vertex $s_{x_i,v}$ has the v-connecting label in $\mathbf{val}(t\langle w_i\rangle)$. Thus, additional $|A_2|$ labels exists in $\mathbf{val}(t\langle w_i\rangle)$. Let v be a vertex in NX_i. By definition there exists a vertex in X_i (say x_j) such that v is adjacent to x_j in G. By Observation 2, vertex $s_{x_j,v}$ has the v-connecting label at w_j. Since v is not in $\mathbf{val}(t\langle w_i\rangle)$, the vertex $s_{x_j,v}$ also has the v-connecting label in $\mathbf{val}(t\langle w_i\rangle)$. Thus, additional $|NX_i|$ labels exists in $\mathbf{val}(t\langle w_i\rangle)$. Finally, by definition x_i has a unique label at w_i. Summarizing all the labels counted so far gives $|\mathbf{Active}(A_1)| + |A_2| + |NX_i| + 1 + q - i = |A| + |NX_i| + 1 - i$. ☐ Since t has Properties 3 and 4 we may assume that the labels 1, 2, and 3 are already considered in the counting of the k labels of t. Since the labels 1, 2, and 3 are not counted in the formula of Observation 3, the next observation follows.

Observation 4. *For* $1 \leq i \leq q$, $k \geq |A| + |NX_i| + 4 - i$.

Observation 5. $k \geq |A| + 3$.

Proof. If $\mathbf{Dead}(A_1) \neq \emptyset$ the claim follows from Observation 4 for $i = 1$. Suppose $\mathbf{Dead}(A_1) = \emptyset$. Let x be any vertex of $\mathbf{Active}(A_1)$. For each vertex v in A_2 the vertex $s_{x,v}$ must have a v-connecting label at a. Thus, so far we have $|A_2|$ different labels at a. Since all the vertices in $\mathbf{Active}(A_1)$ have different labels at a we get $|A_2| + |\mathbf{Active}(A_1)| = |A|$ different labels at a. Since we did not count labels $1, 2,$ and 3, the claim follows. ☐

Observation 6. $\mathbf{labels}(\mathbf{val}(t\langle a\rangle)) \geq |\mathbf{Active}(A_1)| + |A_2| + |B_2|$.

Proof. By Property 7, each vertex $v \in \mathbf{Active}(A_1)$ has a unique label in $\mathbf{val}(t\langle b\rangle)$. Thus there are at least $|\mathbf{Active}(A_1)|$ labels in $\mathbf{val}(t\langle a\rangle)$. Let v be a vertex in A_2 and let u be any vertex in A_1. First assume $u \in \mathbf{Dead}(A_1)$. Since u is dead at a, $s_{u,v}$ must be connected to u in $\mathbf{val}(t\langle a\rangle))$. Now assume $u \in \mathbf{Active}(A_1)$. If $s_{u,v}$ is not connected to u in $\mathbf{val}(t\langle a\rangle)$, then an η-operation above a that connects $s_{u,v}$ to u connects it also to the vertex $x \in \mathbf{Active}(B_1)$ such that $u = \mathbf{mate}(x)$, a contradiction. Hence, in any case $s_{u,v}$ is connected to u and has the v-connecting label in $\mathbf{val}(t\langle a\rangle)$. Thus additional $|A_2|$ labels must exists in $\mathbf{val}(t\langle a\rangle)$. By symmetry, additional $|B_2|$ vertices must exists in $\mathbf{val}(t\langle a\rangle)$. ☐

Since labels 1, 2, and 3 are not counted in the formula of Observation 6 the next observation follows.

Observation 7. $k \geq |\mathbf{Active}(A_1)| + |A_2| + |B_2| + 3$.

Now we start the process of constructing a linear k-expression which defines G''. At each step we show that no more than k labels are used. Moreover, the η-operations added at each step connect special vertices of the form $s_{x,y}$ to x and y, which implies that all edges added in the process belong to G''. Finally, we show in a sequence of observations that for each regular vertex x of G'' the edges which connect x to all its neighbors in G'' exist in the linear cwd-expression that we construct. Thus this expression satisfies the conditions of the lemma.

Let e_1 denote the expression obtained from $t\langle c\rangle$ as follows:

1. Omit all the special vertices of the form $s_{x,y}$ such that both x and y do not occur in $\mathbf{val}(t\langle c\rangle)$.

2. Add immediately above c the following sequence of η-operations: for each

special vertex $s = s_{x,y}$ such that s and x (y) occur in $\mathbf{val}(t\langle c\rangle)$ but are not adjacent in $\mathbf{val}(t\langle c\rangle)$, add an η-operation which connects s and x (y).

OBSERVATION 8. *For each vertex $u \in \mathbf{Dead}(B_1)$, $\mathbf{val}(e_1)$ includes all the edges connecting u to all its neighbors in G''.*

Proof. Let u be a vertex in $\mathbf{Dead}(B_1)$ and let s be a neighbor of u in G''. Clearly, s is a special vertex of the form $s = s_{u,v}$ where v is a regular vertex which is a neighbor of u in G. Since u is dead at $\mathbf{val}(t\langle c\rangle)$, u is adjacent to s in $\mathbf{val}(t\langle c\rangle)$, and therefore the special vertex s is not omitted in step 1 of the construction of e_1. Thus, u is adjacent to s in $\mathbf{val}(e_1)$. ☐

Let e_2 denote the expression obtained from e_1 as follows:

1. For each vertex x such that $\mathbf{val}(e_1)$ includes all the edges connecting x to all its neighbors in G'', add a ρ-operation which renames the label of x to the dead label 1.

2. Omit all the special vertices of the form $s_{x,y}$ such that $x \in \mathbf{Active}(B_1)$ and $y = \mathbf{mate}(x)$.

3. For each regular vertex $u \in \mathbf{Active}(B_1)$ add the following sequence of operations:

3.1. A 1-$\oplus$-operation which introduces $\mathbf{mate}(u)$ with label 3. Note that since t has Property 4, label 3 is not used in $\mathbf{val}(t\langle a\rangle)$, which implies that this label is not used at the root of $\mathbf{tree}(e_1)$.

3.2. A 1-$\oplus$-operation which introduces $s = s_{u,\mathbf{mate}(u)}$ with label 2. Note that since t has Property 4, label 2 is not used in $\mathbf{val}(t\langle a\rangle)$, which implies that this label is not used at the root of $\mathbf{tree}(e_1)$.

3.3. An $\eta_{2,3}$-operation which connects $\mathbf{mate}(u)$ and s.

3.4. An $\eta_{2,\ell}$-operation which connects u and s, where ℓ is the label that u has in $\mathbf{val}(t\langle a\rangle)$.

3.5. A $\rho_{2\to 1}$-operation renaming the label of s to the dead label 1.

3.6. A $\rho_{\ell\to 1}$-operation renaming the label of u to the dead label 1.

3.7. A $\rho_{3\to\ell}$-operation renaming the label of $\mathbf{mate}(u)$ to the label it has in $\mathbf{val}(t\langle a\rangle)$.

OBSERVATION 9. *For each vertex $u \in \mathbf{Active}(B_1)$, $\mathbf{val}(e_2)$ includes all the edges connecting u to all its neighbors in G''.*

Proof. Let $u \in \mathbf{Active}(B_1)$ and let s be a neighbor of u in G''. Clearly, s is a special vertex of the form $s = s_{u,v}$ where v is a regular vertex which is a neighbor of u in G. Suppose $v \neq \mathbf{mate}(u)$. If s is not in $\mathbf{val}(t\langle c\rangle)$ then the η-operation above c in $\mathbf{tree}(t)$ which connects s to u connects it also to $\mathbf{mate}(u)$, a contradiction. Thus, both s and u are in $\mathbf{val}(t\langle c\rangle)$. By step 2 of the construction of e_1, u and s are adjacent in $\mathbf{val}(e_2)$. Suppose $v = \mathbf{mate}(u)$. By step 3.4 of the construction of e_2, s and u are adjacent in $\mathbf{val}(e_2)$. ☐

Let e_3 denote the expression obtained from e_2 by adding the following sequence of operations immediately above the root of $\mathbf{tree}(e_2)$:

1. For each vertex $u \in A_2 \cup B_2$, if there is no u-connecting label in $\mathbf{val}(e_2)$, add a 1-$\oplus$-operation which introduces u with a unique label ℓ_u (distinct from 1, 2, and 3). Otherwise, let ℓ denote the u-connecting label in $\mathbf{val}(e_2)$ (note that we assume that the label ℓ is unique, otherwise we can add ρ-operations which unify all the u-connecting labels to a unique label), and add the following sequence of operations:

1.1. A 1-$\oplus$-operation which introduces u with label 3.

1.2. An $\eta_{3,\ell}$-operation which connects u to all the vertices having a u-connecting label in $\mathbf{val}(e_2)$.

1.3. A $\rho_{\ell\to 1}$-operation renaming label ℓ to the dead label 1.

1.4. A $\rho_{3\to\ell}$-operation renaming the label of u to ℓ.

2. For each special vertex $s = s_{x,y}$ such that both x and y are in $\mathbf{Active}(A_1) \cup A_2 \cup B_2$, add the following sequence of operations:

2.1. A 1-$\oplus$-operation which introduces s with label 2.

2.2. An η_{2,ℓ_x}-operation, which connects s to x, where ℓ_x is the (unique) label of x at that point.

2.3. An η_{2,ℓ_y}-operation, which connects s to y, where ℓ_y is the (unique) label of y at that point.

2.4. A $\rho_{2\to1}$-operation renaming the label of s to the dead label 1.

3. For each regular vertex $u \in B_2 \setminus NX_q$, add a $\rho_{\ell_u\to1}$-operation renaming the label of u to the dead label 1, where ℓ_u is the (unique) label that u has at that point.

OBSERVATION 10. *e_3 is a k-expression, and* $\mathbf{labels}(\mathbf{val}(e_3)) \le |\mathbf{Active}(A_1)| + |NX_q| + |A_2| + 1$.

Proof. The expression e_1 is constructed from $t\langle c\rangle$ without adding new labels. The expression e_2 is constructed from e_1 using the labels of e_1 in addition to the labels 1, 2, and 3 which are already considered in counting the k labels of t. Thus, e_2 is a k-expression.

In the construction of e_3 from e_2 (described above) the highest number of labels used is immediately before the completion of step 2 (which is the same as the number of labels used immediately before the completion of step 1). At that point all the vertices in $\mathbf{Active}(A_1) \cup A_2 \cup B_2$ have unique labels, the vertices in B_1 have label 1, the last special vertex considered has label 2 and all the other special vertices have label 1. Thus the total number of labels used at that point is at most $|\mathbf{Active}(A_1)| + |A_2| + |B_2| + 2$ which, by Observation 7, is less than k. When step 2 is completed the number of labels is reduced by one, since the last special vertex considered gets label 1. After step 3 is completed the number of labels is reduced by $|B_2 \setminus NX_q|$. □

Let $f_0 = e_3$ and for $1 \le i \le q$ let f_i be the expression obtained by adding the following sequence of operations immediately above the root of $\mathbf{tree}(f_{i-1})$:

1. A 1-$\oplus$-operation which introduces $x_{q-(i-1)}$ with a unique label, denoted by $\ell(x_{q-(i-1)})$.

2. For each special vertex $s = s_{x,y}$ such that $x = x_{q-(i-1)}$ and y is in $NX_{q-(i-1)}$ add the following sequence of operations:

2.1. A 1-$\oplus$-operation which introduces s with label 2.

2.2. An $\eta_{2,\ell(x_{q-(i-1)})}$-operation, which connects s to $x_{q-(i-1)}$.

2.3. An η_{2,ℓ_y}-operation, which connects s to y, where ℓ_y is the (unique) label of y at that point.

2.4 A $\rho_{2\to1}$-operation renaming the label of s to the dead label 1.

3. For each regular vertex $u \in NX_{q-(i-1)} \setminus NX_{q-i}$, add a $\rho_{\ell_u\to1}$-operation renaming the label of u to the dead label, where ℓ_u is the (unique) labels that u has at that point.

OBSERVATION 11. *For each vertex $u \in B_2$,* $\mathbf{val}(f_q)$ *includes all the edges connecting u to all its neighbors in G''.*

Proof. Let u be a vertex in B_2 and let s be a neighbor of u in G''. Clearly, s is a special vertex of the form $s = s_{u,v}$ where v is a regular vertex which is a neighbor of u in G. If $v \in \mathbf{Active}(A_1) \cup A_2 \cup B_2$, then the s is connected to u by one of the two η-operations added in steps 2.2 and 2.3 of the construction of e_3. Suppose $v \in B_1$. By Observations 8 and 9, s is connected to v in $\mathbf{val}(e_2)$. Thus, s has a u-connecting label in $\mathbf{val}(e_2)$ and is connected to u in step 1.2 of the construction of e_3. The last case to consider is when v is in $\mathbf{Dead}(A_1)$. In this case $v = x_{q-(i-1)}$ for some $i \in \{1, \dots, q\}$

and u must be in $NX_{q-(i-1)}$. Thus, u (denoted as y) is connected to s in step 2.3 of the construction of f_i. ☐

OBSERVATION 12. *For $0 \le i \le q$, the f_i is a k-expression, and* $\mathbf{labels}(\mathbf{val}(f_i)) \le |\mathbf{Active}(A_1)| + |A_2| + |NX_{q-i}| + 1 + i = |A| + |NX_{q-i}| + 1 - (q-i)$.

Proof. The proof is by induction on i. For $i = 0$ the claim follows from Observation 10, hence assume $i > 0$. It follows by Observation 10 that the number of labels used in e_3 is at most k. The highest number of labels used in the construction of f_i from f_{i-1} is immediately after step 2.1 is completed. At that point the number of labels used is equal to $\mathbf{labels}(\mathbf{val}(f_{i-1}))$ plus one new label for $x_{q-(i-1)}$ plus the label 2 used for introducing the special vertex at step 2.1. By the inductive hypothesis this number is at most $|A| + |NX_{q-(i-1)}| + 3 - (q-(i-1))$ which by Observation 4 is less than k. At the completion of step 2 of the construction of f_i the number of labels is reduced by one since the label 2 is renamed to 1. At the completion of step 3. the number of labels is reduced by $|NX_{q-(i-1)} \setminus NX_{q-i}|$ which gives the claimed formula for $\mathbf{labels}(\mathbf{val}(f_i))$. ☐

Let t' denote the expression obtained from f_q by adding the following sequence of operations immediately above the root of $\mathbf{tree}(f_q)$:

1. For each special vertex $s = s_{x,y}$ such that $x \in \mathbf{Dead}(A_1)$ and $y \in A$ add the following sequence of operations:

1.1. A 1-$\oplus$-operation which introduces s with label 2.

1.2. An η_{2,ℓ_x}-operation, which connects s to x, where ℓ_x is the unique label of x in $\mathbf{val}(f_q)$.

1.3. An η_{2,ℓ_y}-operation, which connects s to y, where ℓ_y is the unique label of y in $\mathbf{val}(f_q)$.

1.4. A $\rho_{2\to 1}$-operation renaming the label of s to the dead label 1.

OBSERVATION 13. *For each vertex $u \in A$,* $\mathbf{val}(t')$ *includes all the edges connecting u to all its neighbors in G''.*

Proof. Let u be a vertex in A and let s be a neighbor of u in G''. Clearly, s is a special vertex of the form $s = s_{u,v}$ where v is a regular vertex which is a neighbor of u in G. We consider the following cases:

Case 1: Suppose $u \in \mathbf{Active}(A_1)$. If $v \in \mathbf{Active}(A_1) \cup A_2 \cup B_2$, then u is connected to s in step 2.2 or step 2.3 of the construction of e_3. If $v \in \mathbf{Active}(B_1)$, then u must be equal to $\mathbf{mate}(v)$ and is connected to s in step 3.3 of the construction of e_2. If $v \in \mathbf{Dead}(A_1)$, then u (denoted as y) is connected to s in step 1.3 of the construction of t'. The last case to consider is when v is in $\mathbf{Dead}(B_1)$. In this case s must occur at c which implies that the η-operation above a in $\mathbf{tree}(t)$ which connects s to u also connects s to the vertex z such that $u = \mathbf{mate}(z)$, a contradiction. Thus, the case when v is in $\mathbf{Dead}(B_1)$ is not possible.

Case 2: Suppose $u \in A_2$. If $v \in \mathbf{Active}(A_1) \cup A_2 \cup B_2$, then u is connected to s in step 2.2 or step 2.3 of the construction of e_3. If $v \in B_1$, then s must have a u-connecting label in $\mathbf{val}(e_2)$ and is connected to u in step 1.2 of the construction of e_3. If $v \in \mathbf{Dead}(A_1)$, then u (denoted as y) is connected to s in step 1.3 of the construction of t'.

Case 3: Suppose $u \in \mathbf{Dead}(A_1)$. If $v \in A$, then u (denoted as x) is connected to s in step 1.2. of the construction of t'. If $v \in \mathbf{Active}(B_1)$, then s must occur at b, which implies that the η-operation above a in $\mathbf{tree}(t)$ which connects s to v also connects s to $\mathbf{mate}(v)$, a contradiction. If $v \in \mathbf{Dead}(B_1)$ then s must occur at c, which is not possible since $u \in \mathbf{Dead}(A_1)$ implies that s must occur at b. The last case to consider is $v \in B_2$. Since $u \in \mathbf{Dead}(A_1)$, $u = x_{q-(i-1)}$ for some $i \in \{1, \ldots, q\}$, and $v \in NX_{q-(i-1)}$. Thus, u is connected to s in step 2.2 of the construction of f_i. □

OBSERVATION 14. *The expression t' defines G''.*

Proof. From the construction of t', it is clear that all the η-operations of t' add edges which belong to G''. To complete the proof we show that all edges of G'' exist in $\mathbf{val}(t')$. Let $e = uv$ be an edge of G''. By definition of G'' one of the two endpoints of e (say u) is a regular vertex. If $u \in A$, then e is present in $\mathbf{val}(t')$ by Observation 13. If $u \in B_1$, then e is present in $\mathbf{val}(t')$ by Observations 8 and 9. If $u \in B_2$, then e is present in $\mathbf{val}(t')$ by Observation 11. □

OBSERVATION 15. *The expression t' is a linear k-expression.*

Proof. Since t has Property 6, a is the unique (> 1)-$\oplus$-operation in t, which implies that $t\langle c\rangle$ is linear. The expression t' is constructed by adding to $t\langle c\rangle$ a sequence of operations which are either η, ρ, or 1-$\oplus$-operations. Thus, t' is a linear expression. To complete the proof we show that at most k labels are used in t'. By Observation 12, the number of labels used in f_q is at most k. The highest number of labels used in the construction of t' from f_q is equal to $\mathbf{labels}(\mathbf{val}(f_q))$ plus one new label which is used to introduce special vertices (with label 2). By Observation 12 this number is at most $|A| + |NX_0| + 1$ which, by Observation 5, is less than k. □

Lemma 16 follows now from Observations 14 and 15. This concludes the proof of Lemma 16.

7.8. Combining the previous results. By combining the previous lemmas we obtain a proof of Theorem 5, which states that $\text{lin-cwd}(G'') \leq \text{cwd}(G'') + 6$ holds if G is a cobipartite graph with minimum degree at least 2.

Proof of Theorem 5. Let t be a k-expression defining G''. By Lemma 10, there exists a $(k+4)$-expression t_1 defining G'' such that t_1 has Property 3. By Lemma 11, there exists a $(k+6)$-expression t_2 defining G'' such that t_2 has Property 4. By Lemma 12, there exists a $(k+6)$-expression t_3 defining G'' such that t_3 has Property 5. By Lemma 14, there exists a $(k+6)$-expression t_4 defining G'' such that t_4 has Property 6. By Lemma 15, there exists a $(k+6)$-expression t_5 defining G'' such that t_5 has Property 7. By Lemma 16, there exists a linear $(k+6)$-expression t' which defines G''. This completes the proof of Theorem 5. □

8. Open Questions. Our results show that the clique-width of a graph cannot be computed in polynomial time unless $P = \text{NP}$, and we are left with the question of the *parameterized complexity* of clique-width: what is the complexity of deciding whether the clique-width of a graph does not exceed a fixed parameter k? In particular, the following questions remain open:

1. Is it possible to recognize graphs of clique-width at most 4 in polynomial time?
2. If k is a fixed constant, is it possible to recognize graphs of clique-width at most k in polynomial time?
3. Is the recognition of graphs of clique-width at most k *fixed-parameter tractable* with k as the parameter? I.e., is it possible to recognize graphs of clique-width at most k in time $O(f(k)n^c)$, where n denotes the size of the given graph, f is a computable function, and c is a constant which does not depend on k.
4. Which graph classes admit clique-width computation in polynomial time?

Obviously, a positive answer to Question 1 is a necessary pre-condition for a positive answer to Question 2, and a positive answer to Question 2 is a necessary pre-condition for a positive answer to Question 3.

9. Acknowledgements. We thank Derek Corneil for introducing us to the problem and for all the inspiring discussions. We also thank Frank Gurski for suggesting the use of Construction 1 for the NP-completeness proof of linear clique-width.

REFERENCES

[1] S. Arnborg, D. G. Corneil, and A. Proskurowski, *Complexity of finding embeddings in a k-tree*, SIAM J. Algebraic Discrete Methods, 8 (1987), pp. 277–284.

[2] H. L. Bodlaender, J. R. Gilbert, H. Hafsteinsson, and T. Kloks, *Approximating treewidth, pathwidth, frontsize, and shortest elimination tree*, J. Algorithms, 18 (1995), pp. 238–255.

[3] R. Boliac and V. Lozin, *On the clique-width of graphs in hereditary classes*, in Algorithms and computation, vol. 2518 of Lecture Notes in Computer Science, Springer Verlag, 2002, pp. 44–54.

[4] A. Brandstädt, F. F. Dragan, H.-O. Le, and R. Mosca, *New graph classes of bounded clique-width*, Theory Comput. Syst., 38 (2005), pp. 623–645.

[5] A. Brandstädt, J. Engelfriet, H.-O. Le, and V. V. Lozin, *Clique-width for 4-vertex forbidden subgraphs*, Theory Comput. Syst., online-first (2006).

[6] D. G. Corneil, M. Habib, J.-M. Lanlignel, B. A. Reed, and U. Rotics, *Polynomial time recognition of clique-width ≤ 3 graphs (extended abstract)*, in Theoretical Informatics, 4th Latin American Symposium (LATIN 2000), G. H. Gonnet, D. Panario, and A. Viola, eds., vol. 1776 of Lecture Notes in Computer Science, 2000, pp. 126–134.

[7] D. G. Corneil and U. Rotics, *On the relationship between clique-width and treewidth*, SIAM J. Comput. To appear. An extended abstract appeared in the Proceedings of Graph-theoretic Concepts in Computer Science (WG 2001), LNCS 2204, pp. 78-90, Springer, 2001.

[8] B. Courcelle and J. Engelfriet, *A logical characterization of the sets of hypergraphs defined by hyperedge replacement grammars*, Math. Systems Theory, 28 (1995), pp. 515–552.

[9] B. Courcelle, J. Engelfriet, and G. Rozenberg, *Context-free handle-rewriting hypergraph grammars*, in Graph-Grammars and their Application to Computer Science, 4th International Workshop, Bremen, Germany, March 5–9, 1990, Proceedings, H. Ehrig, H.-J. Kreowski, and G. Rozenberg, eds., vol. 532 of Lecture Notes in Computer Science, 1991, pp. 253–268.

[10] B. Courcelle, J. Engelfriet, and G. Rozenberg, *Handle-rewriting hypergraph grammars*, J. of Computer and System Sciences, 46 (1993), pp. 218–270.

[11] 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.

[12] ———, *On the fixed parameter complexity of graph enumeration problems definable in monadic second-order logic*, Discr. Appl. Math., 108 (2001), pp. 23–52.

[13] B. Courcelle and M. Mosbah, *Monadic second-order evaluations on tree-decomposable graphs*, Theoret. Comput. Sci., 109 (1993), pp. 49–82.

[14] B. Courcelle and S. Olariu, *Upper bounds to the clique-width of graphs*, Discr. Appl. Math., 101 (2000), pp. 77–114.

[15] B. Courcelle and S. Oum, *Vertex-minors, monadic second-order logic and a conjecture by Seese*, J. Combin. Theory Ser. B, 97 (2007), pp. 91–126.

[16] W. Espelage, F. Gurski, and E. Wanke, *Deciding clique-width for graphs of bounded tree-width*, J. Graph Algorithms Appl., 7 (2003), pp. 141–180.

[17] M. R. Fellows, F. A. Rosamond, U. Rotics, and S. Szeider, *Proving NP-hardness for clique-width I: non-approximability of sequential clique-width*, Electronic Colloquium on Computational Complexity, Report TR05-080, Revision 01 (2005).

[18] F. V. Fomin, P. Heggernes, and J. A. Telle, *Graph searching, elimination trees, and a generalization of bandwidth*, in Fundamentals of computation theory, vol. 2751 of Lecture Notes in Computer Science, Springer Verlag, 2003, pp. 73–85.

[19] A. Glikson and J. A. Makowsky, *NCE graph grammars and clique-width*, in Graph-theoretic concepts in computer science, vol. 2880 of Lecture Notes in Computer Science, Springer Verlag, 2003, pp. 237–248.

[20] F. Gurski and E. Wanke, *Minimizing NLC-width is NP-complete (extended abstract)*, in Graph-theoretic concepts in computer science, vol. 3787 of Lecture Notes in Computer Science, Springer Verlag, 2005, pp. 69–80.

[21] F. Gurski and E. Wanke, *On the relationship between NLC-width and linear NLC-width*, Theoret. Comput. Sci., 347 (2005), pp. 76–89.

[22] F. Gurski and E. Wanke, *Vertex disjoint paths on clique-width bounded graphs*, Theoret. Comput. Sci., 359 (2006), pp. 188–199.

[23] Ö. Johansson, *Clique-decomposition, NLC-decomposition, and modular decomposition—relationships and results for random graphs*, in Proceedings of the Twenty-ninth Southeastern International Conference on Combinatorics, Graph Theory and Computing (Boca Raton, FL, 1998), vol. 132 of Congr. Numer., 1998, pp. 39–60.

[24] M. Karpinski and J. Wirtgen, *On approximation hardness of the bandwidth problem*, Tech. Rep. TR97-041, ECCC, Electronic Colloquium on Computational Complexity, 1997.

[25] N. G. Kinnersley, *The vertex separation number of a graph equals its path-width*, Information Processing Letters, 42 (1992), pp. 345–350.

[26] D. Kobler and U. Rotics, *Edge dominating set and colorings on graphs with fixed clique-width*, Discr. Appl. Math., 126 (2003), pp. 197–221.

[27] V. Lozin and D. Rautenbach, *Chordal bipartite graphs of bounded tree- and clique-width*, Discrete Math., 283 (2004), pp. 151–158.

[28] ———, *On the band-, tree-, and clique-width of graphs with bounded vertex degree*, SIAM J. Discrete Math., 18 (2004), pp. 195–206.

[29] J. A. Makowsky and U. Rotics, *On the clique-width of graphs with few P_4's*, Internat. J. Found. Comput. Sci., 10 (1999), pp. 329–348.

[30] S. Oum, *Approximating rank-width and clique-width quickly*, ACM Transactions on Algorithms, 5 (2008).

[31] S. Oum and P. Seymour, *Approximating clique-width and branch-width*, J. Combin. Theory Ser. B, 96 (2006), pp. 514–528.

[32] D. Seese, *The structure of the models of decidable monadic theories of graphs*, Annals of Pure and Applied Logic, 53 (1991), pp. 169–195.

[33] J.-M. Vanherpe, *Clique-width of partner-limited graphs*, Discrete Math., 276 (2004), pp. 363–374.

[34] E. Wanke, *k-NLC graphs and polynomial algorithms*, Discr. Appl. Math., 54 (1994), pp. 251–266. Efficient algorithms and partial k-trees.