Proceedings Paper:
Adler, I, Kante, MM and Kwon, O-J (2014) Linear Rank-Width of Distance-Hereditary Graphs. In: Kratsch, D and Todinca, I, (eds.) Graph-Theoretic Concepts in Computer Science. 40th International Workshop (WG 2014), 25-27 Jun 2014, Nouan-le-Fuzelier, France. Lecture Notes in Computer Science (8747). Springer , pp. 42-55. ISBN 978-3-319-12339-4

https://doi.org/10.1007/978-3-319-12340-0_4

Linear Rank-Width and Linear Clique-Width of Trees

Isolde Adler[1*] and Mamadou Moustapha Kanté[2**]

[1] Institut für Informatik, Goethe-Universität, Frankfurt, Germany.
iadler@informatik.uni-frankfurt.de
[2] Clermont-Université, Université Blaise Pascal, LIMOS, CNRS, France.
mamadou.kante@isima.fr

Abstract We show that for every forest T the linear rank-width of T is equal to the path-width of T, and we show that the linear clique-width of T equals the path-width of T plus two, provided that T contains a path of length three. It follows that both linear rank-width and linear clique-width of forests can be computed in linear time. Using our characterization of linear rank-width of forests, we determine the set of minimal excluded acyclic vertex-minors for the class of graphs of linear rank-width at most k.

1 Introduction

Rank-width [29] is a graph parameter introduced by Oum and Seymour with the goal of efficient approximation of the *clique-width* [9] of a graph. *Linear rank-width* can be seen as the linearized variant of rank-width, similar to path-width, which can be seen as the linearized variant of tree-width. While path-width is a well-studied notion, much less is yet known about linear rank-width. Indeed, any graph of k-bounded path-width has k-bounded linear rank-width, but conversely the difference is unbounded. For example, the class of all complete (bipartite) graphs has linear rank-width at most 1, but unbounded path-width. *Linear clique-width*, a linearized version of clique-width, was introduced independently by several authors when studying the computational complexity of clique-width (see for instance the works by Gurski et al. [11,13,14,15,16], and the paper [27] by Lozin and Rautenbach). The computation of the linear clique-width of some graph classes have been investigated by Heggernes et al. [17,18,19]. Linear rank-width is equivalent to linear clique-width in the sense that any graph class has bounded linear clique-width if and only if it has bounded linear rank-width.

Computing linear rank-width is NP-complete in general. In fact, it is proved in [11] that computing linear clique-width is NP-complete and one can easily reduce the computation of linear clique-width to the computation of linear rank-width. Moreover, very little is known about efficient computation of linear rank-width on restricted graph classes. The only known results are for special types of graphs, such as for complete (bipartite) graphs, and linear clique-width is known to be polynomial time computable on *thickend paths* [19] and *k-path powers* [17]. Even for the very natural class of forests efficient computability was open. In contrast, many classes are known that allow efficient computation of path-width [3,4,5,10,12,24,28,31].

In this paper, we provide the first non-trivial graph class on which linear rank-width can be computed in polynomial (even linear) time. We prove

Theorem 1 *Linear rank-width and linear clique-width of forests can be computed in linear time.*

* Supported by the German Research Council, Project GalA, AD 411/1-1.
** Supported by the French Agency for Research under the DORSO project.

We obtain Theorem 1 as a corollary of the following theorems.

Theorem 2 *The linear rank-width of any forest equals its path-width.*

Theorem 3 *Let T be a forest. If T contains a path of length* 3*, then* $\mathrm{lcw}(T) = \mathrm{pw}(T) + 2$*. Otherwise,* $\mathrm{lcw}(T) = \mathrm{pw}(T) + 1$.

While it was known that the class of all trees has unbounded linear rank-width (see [12] for a combinatorial proof) and unbounded path-width, Theorem 2 is somewhat surprising, because it actually equates the two structurally very different parameters.

It is known that the linear clique-width of any graph is bounded by its path-width plus 2 [11]. Since linear rank-width is bounded by linear clique-width, the same bound carries over to linear rank-width. We show that the linear rank-width of any graph is bounded by its path-width. This is not hard to prove, but it seems it was not written down yet. For forests we show that the converse holds, too. Our proof uses the characterization of path-width by the cops and invisible robber game [23]. Given an ordering of the vertices of a forest T witnessing the linear rank-width of T, we construct a winning strategy for the cops. Here it is not sufficient for the cops to search the vertices according to the given ordering, but a more involved strategy yields the result. Indeed, our proof method is constructive in the sense that it shows how to transform the given ordering into a winning strategy for the cops (and a path decomposition).

It is known that the (linear) rank-width does not increase when taking *vertex-minors*, and, given k, the set of minimal excluded vertex-minors for the class of graphs of rank-width at most k is known to be computable [20]. However, until now, explicit sets of minimal excluded vertex-minors are only known for circle graphs [7], distance-hereditary graphs [20], and for graphs of linear rank-width at most one [1]. For graphs of linear rank-width at most k, some minimal excluded vertex-minors were established in [22]. Using Theorem 2, we determine the set of minimal excluded acyclic vertex-minors for linear rank-width k. It turns out that they coincide with the minimal excluded minors for graphs of path-width at most k that are acyclic [32].

Summary. Section 2 introduces the terminology and the notions of linear rank-width, path-width and the cops and invisible robber game. In Section 3 we prove that linear rank-width and path-width coincide on forests (Theorem 2), and in Section 4 we prove Theorem 3 characterizing the linear clique-width of forests. In Section 5 we give the set of minimal excluded acyclic vertex-minors for the class of graphs of linear rank-width k, and we conclude with Section 6.

2 Preliminaries

For a set A we denote the power set of A by 2^A. We let $A \setminus B := \{x \in A \mid x \notin B\}$ denote the *difference* of two sets A and B. For a subset X of a ground set A let $\overline{X} := A \setminus X$. For two sets A and B let $A \Delta B := (A \setminus B) \cup (B \setminus A)$ denote the *symmetric difference* of A and B. For an integer $n > 0$ we let $[n] := \{1, \dots, n\}$.

In this paper, graphs are finite, simple and undirected, unless stated otherwise. Let G be a graph. We denote the vertex set of G by $V(G)$ and the edge set by $E(G)$. We regard edges as two-element subsets of $V(G)$. For a vertex $v \in V(G)$ we let $N_G(v) := \{u \in V(G) \mid u \neq v,\ \{v, u\} \in E(G)\}$ denote the set of *neighbors* of v (in G). The *degree* of v (in G) is $\deg_G(v) := |N_G(v)|$. A partition of $V(G)$ into two sets X and Y with $X \mathbin{\dot\cup} Y = V(G)$ is called a *cut* in G. We denote it by (X, Y). A *tree* is a connected, acyclic graph. A *leaf* of a tree is a vertex of degree one. A *path* is a tree where every vertex has degree at most two. The *length* of a path is the number of its edges. The *distance* between two vertices $u, v \in V(G)$ is the length of a shortest path from u to v. A *rooted tree* is a tree with a distinguished vertex r, called the *root*. The *height* of a rooted tree is the maximal length of a path from the root to a leaf (counted in terms of edges). Let T be a rooted tree with root r. Let $v \in V(T)$. The tree T^v is the subtree of T induced by those vertices $u \in V(T)$ such that the path from r to u contains v. For a rooted tree T it is sometimes convenient to orient the edges of T in the direction away from the root, thus obtaining an *oriented tree*.

Path-width. A *path decomposition* of a graph G is a pair (P, B), where P is a path and $B = (B_t)_{t \in V(P)}$ is a family of subsets $B_t \subseteq V(G)$, satisfying

1. For every $v \in V(G)$ there exists a $t \in V(P)$ such that $v \in B_t$.
2. For every $e \in E(G)$ there exists a $t \in V(P)$ such that $e \subseteq B_t$.
3. For every $v \in V(G)$ the set $\{t \in V(P) \mid v \in B_t\}$ is connected in P.

The *width* of a path decomposition (P, B) is defined as $\mathrm{w}(P, B) := \max\{|B_t| \mid t \in V(P)\} - 1$. The *path-width* of G is defined as

$$\mathrm{pw}(G) := \min\{\mathrm{w}(P, B) \mid (P, B) \text{ is a path decomposition of } G\}.$$

Paths have path-width ≤ 1. Indeed, the graphs of path-width ≤ 1 are precisely the disjoint unions of *caterpillars*, i.e. of the graphs that contain a path P such that every vertex has distance at most one to some vertex of P. There is no finite upper bound on the path-width of trees. Indeed, the rooted binary tree T_h of height h satisfies $\mathrm{pw}(T_h) = \lceil h/2 \rceil$ [30].

A path decomposition (P, B) of G is *small* if any two distinct vertices $t, t' \in V(P)$ satisfy $B_t \not\subseteq B_{t'}$. The following lemma is not hard to prove.

Lemma 4 *Any graph G has a small path decomposition of width* $\mathrm{pw}(G)$. □

Linear rank-width. For sets R and C an (R, C)*-matrix* is a matrix where the rows are indexed by elements in R and columns indexed by elements in C. (Since we are only interested in the rank of matrices, it suffices to consider matrices up to permutations of rows and columns.) For an (R, C)-matrix M, if $X \subseteq R$ and $Y \subseteq C$, we let $M[X, Y]$ be the submatrix of M where the rows and the columns are indexed by X and Y respectively. If M is an (R, C)-matrix and when the context is clear we will identify the row indexed by $x \in R$ with x (similarly for the column indexed by $y \in C$); hence we will say for instance that a subset X of R is a basis for the rows of M if the rows indexed by X form a basis for the rows of M and similarly for other linear algebra terminologies involving rows (or columns).

Let A_G be the adjacency $(V(G), V(G))$-matrix of G. For a graph G, let $v_1, \ldots, v_n$ be a linear ordering of $V(G)$. Every index $i \in [n]$ induces a cut $(X_i, \overline{X_i})$, where $X_i = \{v_1, \ldots, v_i\}$. The *cutrank* of the ordering $v_1, \ldots, v_n$ is defined as

$$\mathrm{cutrk}_G(v_1, \ldots, v_n) := \max\{\mathrm{rk}(A_G[X_i, \overline{X_i}]) \mid i \in [n]\}.$$

The *linear rank-width* of G is defined as

$$\mathrm{lrw}(G) := \min\{\mathrm{cutrk}_G(v_1, \ldots, v_n) \mid v_1, \ldots, v_n \text{ is a linear ordering of } V(G)\}.$$

Disjoint unions of caterpillars have linear rank-width ≤ 1. Ganian [12] gives an alternative characterization of the graphs of linear rank-width ≤ 1 as *thread graphs*. In addition, he proves that there is no finite upper bound on the linear rank-width of trees.

The cops and invisible robber game We now introduce the *cops and invisible robber game* characterizing path-width. Let G be a graph and let $k \geq 0$ be an integer. The *cops and invisible robber game* on G (with game parameter k) is played by two players, the *cop player* and the *robber player*, on the graph G. The cop player controls k cops and the robber player controls the robber. Both the cops and the robber move on the vertices of G. Some of the cops move to at most k vertices and the robber stands on a vertex r not occupied by the cops. At all times, the robber is invisible to the cops. Initially, no cops occupy vertices and the robber chooses a vertex to start playing. In each move, some of the cops fly in helicopters to at most k new vertices. During the flight, the robber sees which position the cops are approaching and before they land she quickly tries to escape by running arbitrarily fast along paths of G to a vertex r', not being allowed to run through a vertex occupied by a cop. Hence, if $X \subseteq V(G)$ is the cops' position, the robber stands on $r \in V(G) \setminus X$, and after the flight, the cops occupy the set $Y \subseteq V(G)$, then the robber can run to any vertex r' within the connected component of $G \setminus (X \cap Y)$ containing r. The cops win if they land a cop via helicopter on the vertex occupied by the robber. The robber wins if she can always elude capture. A *play* is a sequence of cop positions $X_0, X_1, X_2, \ldots$ with $X_0 := \emptyset$ and $|X_i| \leq k$ for all i. At each step of a play, we can describe the set of *cleared* vertices as follows. At the position X_0, the set of cleared vertices is $A_0 := \emptyset$. After the cops' move to X_i (for $i > 0$), the set of cleared vertices is

$$\begin{aligned} A_i := (A_{i-1} \cup X_i) \setminus \{r \in V(G) \mid & \text{ there is a path from } V(G) \setminus A_{i-1} \\ & \text{to } r \text{ in } G \setminus (X_{i-1} \cap X_i)\}. \end{aligned}$$

Winning strategies are defined in the usual way. The *invisible cop-width* of G, $\mathrm{icw}(G)$, is the minimum number of cops having a winning strategy on G.

A winning strategy for the cops is *monotone*, if for any play $X_1, X_2, X_3, \ldots$ played according to the strategy, the sets $A_0, A_i, A_2, \ldots$ form a non-decreasing sequence (with respect to $\subseteq$). The *monotone invisible cop-width* of G, $\mathrm{monicw}(G)$, is the minimum number of cops having a monotone winning strategy on G.

Theorem 5 ([2,26]) *Any graph G satisfies* $\mathrm{pw}(G) + 1 = \mathrm{icw}(G) = \mathrm{monicw}(G)$. □

3 Linear Rank-Width and Path-Width

In this section we prove that on forests, linear rank-width and path-width coincide. Due to space constraints some proofs are omitted.

Lemma 6 *Any graph G satisfies* $\mathrm{lrw}(G) \leq \mathrm{pw}(G)$.

Definition 7 *Let G be a graph and let (X, Y) be a cut in G. A vertex $x \in X$ is a* standard vertex (of the cut) *if x has exactly one neighbor in Y.*

Fact 8 *Let T be a tree and let (X, Y) be a cut in T.*

1. *Any two distinct rows of $A_T[X, Y]$ have at most one common non-zero position.*
2. *Let $B \subseteq X$ be a basis of the rows of M. A vertex $x \in X \setminus B$ cannot be generated by less than $|N_T(x) \cap Y|$ elements of B.*

Lemma 9 (Spanning dependent vertices) *Let T be a tree and let (X, Y) be a cut in T. Let $B \subseteq X$ be a basis of the row space of $A_T[X, Y]$. For $x \in X \setminus B$ with $N_T(x) \cap Y \neq \emptyset$ let $B' \subseteq B$ be the (unique) minimal subset of B spanning x.*

*We let T', called B-*basic *tree of x, be the bipartite subgraph of T with vertex set $V(T') = X' \dot{\cup} Y'$, where $X' := B' \cup \{x\}$ and $Y' := N_T(B' \cup \{x\}) \cap Y$, and with edge set $E(T') := \{\{u, v\} \in E(T) \mid u \in X',\ v \in Y'\}$. Then*

1. *T' is a tree.*
2. *The leaves of T' are standard vertices in X.*
3. *The vertices in Y' have degree two in T'.*
4. *Choose x to be the root of T' and orient the edges of T' away from the root. Let $b\colon B' \to Y'$ where for every $z \in B'$ we let $b(z)$ be the predecessor of z in T' oriented. Then b is a bijection, and hence $|Y'| = |B'|$.*

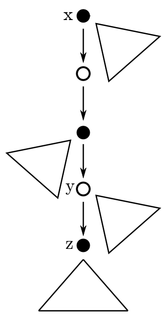

Figure 1. The tree T' in the proof of Lemma 9. Black vertices are in X', white vertices in Y'.

Lemma 10 (Clearing dependent vertices) *Under the conditions of Lemma 9, suppose that in the $(k+1)$-cops and robber game on T the cops have cleared all vertices in $X \setminus \{x\}$ and the game is in a position where at most k cops are occupying vertices. Furthermore, assume that exactly $|B'|$ cops are occupying vertices of T', and in addition, for each vertex $b \in B'$, either b is occupied by a cop, or $N_T(b) \cap Y$ is occupied by cops. Then there is a sequence of moves of $|B'|+1$ cops, involving only the cops on vertices of T' plus one additional cop, that ends in a position, where*

1. *the vertices in $X \cup V(T') \setminus \{x\}$ are cleared,*
2. *all vertices in $N_T(x) \cap Y$ are occupied,*
3. *exactly $|B'|$ cops occupy vertices of T', and*
4. *the set $N_T(B') \cap Y$ is occupied by cops.*

Theorem 11 *Any forest T satisfies* $\mathrm{pw}(T) \leq \mathrm{lrw}(T)$.

Proof. We may assume that T is a tree. Let $v_1, \ldots, v_n$ be a linear ordering of $V(T)$ witnessing $k := \mathrm{lrw}(T)$. For $i \in [n]$ let $X_i := \{v_1, \ldots, v_i\}$ and $Y_i := \{v_{i+1}, \ldots, v_n\}$, and let M_i be $A_T[X_i, Y_i]$.

We describe a strategy for $k+1$ cops in the invisible robber and cops game. The strategy follows the linear ordering of $V(T)$. For each new vertex v_i that has to be cleared, we describe a *transition* – a finite sequence of cop moves to make sure that v_i is cleared. After the ith transition, the following invariants hold.

1. Every vertex in X_i is cleared.
2. There is a basis $B_i \subseteq X_i$ of the rows of M_i such that each $b \in B_i$ satisfies:
   b is occupied by a cop or $N_T(b) \cap Y_i$ is occupied by cops, and no vertex in the set $X_i \setminus B_i$ is occupied by a cop.

3. The cops occupy exactly $|B_i|$ vertices.

The first $\leq k$ transitions simply consist in placing cops on the vertices $v_1, \ldots, v_\ell$, with $\ell \leq k$, successively, where ℓ is the greatest index $i \leq k$ such that the rank of M_i is equal to i. Obviously, after each such transition the invariants hold.

Suppose we have completed the ith transition, and we want to make the $(i+1)$st transition. Moving from M_i to M_{i+1}, the following cases can occur.

(a) In M_{i+1}, the new vertex v_{i+1} is in the span of B_i.
(b) In M_{i+1}, the new vertex v_{i+1} is linearly independent of B_i.

Observe that B_i can span the rows of M_{i+1}, but may be linearly dependent in M_{i+1}. If it is linearly dependent in M_{i+1}, then the size of a maximum linearly independent subset of B_i is $|B_i| - 1$, because deleting a column can only decrease the rank by one.

Claim 1: If the size of a maximum linearly independent subset of B_i in M_{i+1} is $|B_i| - 1$, then there exists a vertex $v_N \in N_T(v_{i+1}) \cap B_i$ such that $B_i \setminus \{v_N\}$ is a maximum linearly independent subset of B_i in M_{i+1}.

Proof of the Claim: If B_i is linearly dependent in M_{i+1} and linearly independent in M_i, there exists a row of M_i corresponding to a vertex $u \in B_i$ that is generated by $B_i \setminus \{u\}$ and that has a 1 at the column corresponding to v_{i+1}, and hence $u \in N_T(v_{i+1})$. $\dashv$

We will complete the $(i+1)$st transition in such a way that the new basis B_{i+1} of the row space of M_{i+1} contains a basis of the rows of M_{i+1} corresponding to B_i, together with the vertex v_{i+1}, if v_{i+1} is linearly independent of B_i in M_{i+1}. For this, let $v_N \in N_T(v_{i+1}) \cap B_i$ be as in Claim 1. If v_{i+1} is in the span of B_i, we let $B_{i+1} := B_i \setminus v_N$. Otherwise, we let $B_{i+1} := (B_i \setminus v_N) \cup \{v_{i+1}\}$. Obviously, B_{i+1} is a basis of M_{i+1}.

For each v spanned by B_{i+1} let T_v denote the B_{i+1}-basic tree of v. The following follows from the fact that T is a tree and the vertex v_N is adjacent to v_{i+1}.

Claim 2: let $v_N \in N_T(v_{i+1}) \cap B_i$ be as in Claim 1.

(i) The B_{i+1}-basic tree of v_N does not contain v_{i+1}.
(ii) $V(T_{v_{i+1}}) \cap V(T_{v_N}) = \emptyset$ and there is no edge other than $\{v_N, v_{i+1}\}$ between a vertex of $T_{v_{i+1}}$ and a vertex of T_{v_N}. $\dashv$

We identify two cases, depending on whether a cop occupies v_{i+1}.

Case 1. After the ith transition, v_{i+1} is not occupied by a cop.
Then by the inductive invariant (1), the set $N_T(v_{i+1}) \cap X_{i+1} = N_T(v_{i+1}) \cap X_i$ is occupied by cops, and hence $N_T(v_{i+1}) \cap X_i \subseteq B_i$ by the inductive invariant (2).

Case 1.1 Vertex v_{i+1} is in the span of B_i in M_{i+1}.
If v_{i+1} has no neighbors in Y_{i+1}, then we use the $(k+1)$st cop to step on v_{i+1} and remove the cop again. Otherwise, let T' be the B_{i+1}-basic tree of v_{i+1}, and let $B' \subseteq B_{i+1}$ be the minimal subset of B_{i+1} spanning v_{i+1}. Since T has no cycles, $V(T') \cap \big(N_T(v_{i+1}) \cap X_{i+1}\big) = \emptyset$. Hence we can use Lemma 10 to move to $N_T(v_{i+1}) \cap Y_{i+1}$ with at most $|B'| + 1 \leq k+1$ cops, ending in a position where at most k cops are on $V(T)$. Since $N_T(v_{i+1}) \cap X_{i+1}$ is occupied by cops, we can use the $(k+1)$st cop to step on v_{i+1} and then lift the $(k+1)$st cop up again, thus clearing v_{i+1}. We have then cleared X_{i+1}.

It remains to check conditions (2) and (3). By the inductive hypothesis invariant, (2) is already satisfied, and if B_i is linearly independent in M_{i+1}, condition (3) is also satisfied. So assume B_i is linearly dependent in M_{i+1}. If v_N does not have a neighbor in Y_{i+1} we can remove safely the cop from v_N. Otherwise, if it has a neighbor in Y_{i+1}, we can use Lemma 10 to move to $N_T(v_N) \cap Y_{i+1}$, and we then lift up the cop from v_N. By Claim 2, we can do it safely. In this way, we end the transition with a position of $|B_{i+1}|$ cops on $V(T)$. This follows from Lemma 10(3) and Claim 2. Hence all three invariants are satisfied.

Case 1.2. Vertex v_{i+1} is not in the span of B_i in M_{i+1}.
If v_N has no neighbors in Y_{i+1}, we place the $(k+1)$st cop on v_{i+1} (v_{i+1} is not already occupied by a cop) and we then remove the cops from v_N. After these moves, at most k cops are occupying vertices.

Now, if v_N has a neighbor in Y_{i+1}, take the B_{i+1}-basic tree T_{v_N} of v_N and use Lemma 10 to move cops in $V(T_{v_N}) \setminus \{v_N\}$ to $N_T(v_N) \cap Y_{i+1}$. Claim 2 guarantees the safety of these moves. After these moves, at most k cops are occupying vertices. If v_{i+1} was occupied by a cop, then remove the cop that is still occupying the vertex v_N. If v_{i+1} was not occupied by a cop, then we place the $(k+1)$st cop on v_{i+1} and remove the cop that occupy the vertex v_N. After these moves, v_{i+1} is cleared and since we did not recontaminate X_i, X_{i+1} is cleared. Moreover, exactly $|B_{i+1}|$ vertices of T are occupied by cops (Lemma 10(3) and Claim 2), and since the other cops are not moved, invariant (2) is satisfied. The three invariants are hence satisfied.

Case 2. After the ith transition, v_{i+1} is occupied by a cop.
By the inductive invariant (1), each vertex $b \in N_T(v_{i+1}) \cap X_{i+1} = N_T(v_{i+1}) \cap X_i$ is cleared, hence either b is occupied by a cop, or $N_T(b) \cap Y_{i+1}$ is occupied by cops.

Case 2.1. Vertex v_{i+1} is in the span of B_i in M_{i+1}.
For every $b \in \{v_N, v_{i+1}\}$ such that $V(T_b) \cap Y_{i+1}$ contains an unoccupied vertex, we use Lemma 10 to move cops in $V(T_b) \setminus \{b\}$ to $V(T_b) \cap Y_{i+1}$. This is possible, because the B_{i+1}-basic trees involved are pairwise disjoint and pairwise connected via v_{i+1} only (Claim 2). After that, we remove the cops occupying vertices in $\{v_N, v_{i+1}\}$. Since by induction, the cop moves are monotone, we can conclude that the three inductive invariants are satisfied.

Case 2.2. Vertex v_{i+1} is not in the span of B_i in M_{i+1}.
If $V(T_{v_N}) \cap Y_{i+1}$ contains an unoccupied vertex, we use Lemma 10 to move cops in $V(T_{v_N}) \setminus \{v_N\}$ to $V(T_{v_N}) \cap Y_{i+1}$. After that, we remove the cop occupying the vertex v_N. Since by induction, the cop moves are monotone, we can conclude that the three inductive invariants are satisfied. □

Note that the analogous statement of Theorem 11 fails for C_3, the cycle of length three. While $\mathrm{lrw}(C_3) = 1$, we have $\mathrm{pw}(C_3) = 2$.

Theorem 2 now follows from Lemma 6 and Theorem 11. Theorem 2 combined with [10] gives the following as a corollary.

Theorem 12 *There is a linear time algorithm that computes the linear rank-width of any forest, and an ordering of its vertex set V witnessing its linear rank-width can be computed in time $\mathcal{O}(|V| \cdot \log |V|)$.*

Example 13 *Let T be the graph shown in Figure 2. The ordering b, a, c, d, e is a witness for $\mathrm{lrw}(T) \leq 1$. The strategy for two cops according to the proof of Theorem 11 is as follows: the first cop moves to b and then the second cop moves to a and remains there. Now the first cop moves to c, d, e in this ordering.*

Example 14 *The tree T in Figure 3 satisfies $\mathrm{lrw}(T) = 2$. The given ordering (attached to the vertices) witnesses $\mathrm{lrw}(T) \leq 2$. The strategy for three cops according to Theorem 11 is* $\{1\}, \{1,2\}, \{2,3\}, \{2,4\}, \{4,5\}, \{4,5,6\}, \{4,6,7\}, \{4,8\}, \{8,9\}, \{8,9,10\}, \{8,10,11\}, \{8,10,12\}, \{8,12,14\}, \{8,13,14\}, \{8,14,15\}, \{8,16\}, \{8,16,17\}, \{8,17,18\}, \{8,17,19\}, \{8,19,20\}, \{19,20,21\}, \{21,22\}$.

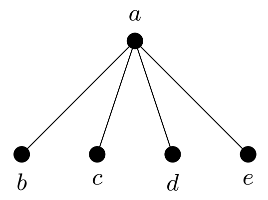

Figure 2. The tree of Example 13.

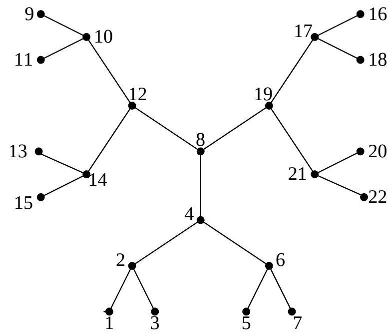

Figure 3. The tree of Example 14.

4 Linear Clique-Width

In this section we prove Theorem 3, characterizing the linear clique-width of forests in terms of their path-width. It follows that the linear clique-width of forests is linear time computable. In [11] it is proved that the linear clique-width of a graph is at most its path-width plus 2. We prove that for forests containing a path of length three, this upper bound is also a lower bound.

Let us recall the definition of linear clique-width [11,15,27]. Let k be a positive integer. A *k-labeled* graph is a pair (G, γ) where G is a graph and $\gamma : V(G) \to [k]$ is a mapping; we will also denote it by $(V(G), E(G), \gamma)$. The k-labeled graph consisting of a single vertex labeled by $i \in [k]$ is denoted by $(\mathbf{i}, \gamma_{\mathbf{i}})$. The set LIN-CW$_k$ of k-labeled graphs is defined inductively with the following operations.

1. For each $i \in [k]$, $(\mathbf{i}, \gamma_{\mathbf{i}})$ is in LIN-CW$_k$.
2. If $i, j \in [k]$ and (G, γ) is in LIN-CW$_k$, then $(\rho_{i\to j}(G), \gamma)$ is in LIN-CW$_k$ and denotes the k-labeled graph $(V(G), E(G), \gamma')$ with
$$\gamma'(x) := \begin{cases} \gamma(x) & \text{if } \gamma(x) \neq i, \\ j & \text{otherwise.} \end{cases}$$
3. If $i, j \in [k]$, $i \neq j$, and (G, γ) is in LIN-CW$_k$, then $(\eta_{i,j}(G), \gamma)$ is in LIN-CW$_k$ and denotes the k-labeled graph $(V(G), E', \gamma)$ with
$$E' := E(G) \cup \{\{x, y\} \mid \gamma(x) = i \text{ and } \gamma(y) = j\}\,.$$
4. If $i \in [k]$ and (G, γ) is in LIN-CW$_k$, then $(G \oplus \mathbf{i}, \gamma')$ is in LIN-CW$_k$ and denotes the graph $(V(G) \cup \{z\}, E(G), \gamma')$ where $z \notin V(G)$ and
$$\gamma'(x) := \begin{cases} \gamma(x) & \text{if } x \in V(G), \\ i & \text{otherwise.} \end{cases}$$

An expression built with the operations $\mathbf{i}, \rho_{i\to j}, \eta_{i,j}$ and $\oplus$ according to the definition of LIN-CW$_k$ is called a *linear k-expression*. The *linear clique-width* of a graph G, denoted by $\mathrm{lcw}(G)$, is the minimum k such that G is isomorphic to a graph in LIN-CW$_k$ (after forgetting the labels). It is worth noticing that if H is an induced subgraph of G, then $\mathrm{lcw}(H) \leq \mathrm{lcw}(G)$. Moreover, any linear k-expression t defining a graph G defines a linear ordering of $V(G)$ witnessing the ordering in which the vertices of G appears in t.

Lemma 15 ([8,11]) *Any graph G satisfies* $\mathrm{lcw}(G) \leq \mathrm{pw}(G) + 2$. □

The proofs of Lemmas 17 and 18 are omitted due to space constraints. For the proof of Lemma 18 we use the following lemma, proved in [10, Theorem 3.1].

Lemma 16 *Let T be a tree and let $k \geq 1$ be an integer. Then* $\mathrm{pw}(T) \leq k$ *if and only if for all $v \in V(T)$ at most two of the trees in $T \setminus v$ have path-width k and all others have path-width less than k.* □

Lemma 17 *Let T be a tree obtained from three trees T_1, T_2 and T_3 by adding a new vertex r adjacent to exactly one vertex in each of the three trees. If* $\mathrm{lcw}(T_i) = k$ *for each $i \in \{1, 2, 3\}$, then* $\mathrm{lcw}(T) \geq k + 1$.

Lemma 18 *Any forest T containing a path of length three satisfies* $\mathrm{lcw}(T) \geq \mathrm{pw}(T) + 2$.

Proof of Theorem 3. We can assume without loss of generality that T is a tree. The first statement follows from Lemmas 15 and 18. For the second statement, if T does not contain a path of length three, then it is a star. Since stars with at least one edge have linear clique-width 2 and path-width 1, we can conclude that $\mathrm{lcw}(T) = \mathrm{pw}(T) + 1$. □

5 Minimal Excluded Acyclic Vertex-Minors

As an application, in this section we identify the minimal excluded *acyclic* vertex-minors for linear rank-width k. For this result we use both Lemma 16 and the fact that linear rank-width and path-width coincide on trees.

For a graph G and a vertex x of G, the *local complementation at x* of G consists in replacing the subgraph induced on the neighbors of x by its complement. The resulting graph is denoted by $G * x$. If H can be obtained from G by a sequence of local complementations, then G and H are called *locally equivalent*. A graph H is called a *vertex-minor* of a graph G if H is isomorphic to a graph obtained from G by applying a sequence of local complementations and deletions of vertices. The graph H is a *proper vertex-minor* of G if H is a vertex-minor of G and $|V(H)| < |V(G)|$. A graph G is a *minimal excluded vertex-minor* for the class of graphs of linear rank-width k, if $\mathrm{lrw}(G) > k$ and $\mathrm{lrw}(H) \leq k$ for all proper vertex-minors H of G. It is known that for fixed k, the set of minimal excluded vertex-minors for the class of graphs of linear rank-width at most k is finite [21]. For $k = 1$, the set of minimal excluded vertex-minors consists of three graphs [1]. For $k \geq 2$, a double-exponential lower bound on the number of minimal excluded vertex-minors is known [22] . See also [6,20] for more information on vertex-minors.

We say that a graph G is a *minimal excluded acyclic vertex-minor* for the class of graphs of linear rank-width k, if G is acyclic and every proper acyclic vertex-minor of G has linear rank-width less than k. Note that a minimal excluded acyclic vertex-minor may not be a minimal excluded vertex minor. For example, let R_3 be the the tree obtained from the star with three leaves by subdividing each edge once (cf. Figure 4). Then R_3 is a minimal excluded acyclic vertex-minor for the class of graphs of linear rank-width at most 1, but it contains the net graph (i. e. the graph obtained from a triangle by adding three pendant vertices, one to each of the vertices of the triangle) shown in Figure 4 as a proper vertex minor, which in turn is a minimal excluded vertex-minor for the class of graphs of linear rank-width at most one [1].

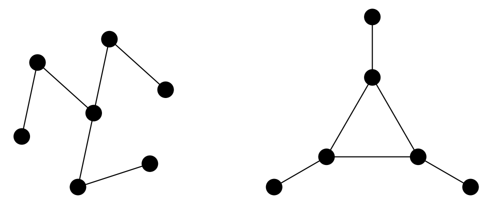

Figure 4. The subdivided 3-star R_3, and the net graph.

We now determine the set of pairwise not locally equivalent minimal excluded acyclic vertex-minors for linear rank-width k. Due to minimality, the minimal excluded (acyclic) vertex-minors for linear rank-width k are necessarily connected. Let $\mathcal{H}_1 := \{R_3\}$. For $k \geq 2$, let $\mathcal{H}_k$ be the set of (pairwise non isomorphic) trees obtained by taking a new vertex r and three trees in $\mathcal{H}_{k-1}$, and by linking this new vertex to one vertex in each of these three trees. Notice that two trees in $\mathcal{H}_k$ have the same size.

Lemma 19 *Let $k \geq 1$ be an integer. Every tree of linear rank-width $k+1$ contains a tree in $\mathcal{H}_k$ as a vertex-minor.*

Theorem 20 *For each $k \geq 1$, the set $\mathcal{H}_k$ is the set of minimal excluded acyclic vertex-minors for linear rank-width k.*

Proof. One can prove by induction, by using Theorem 2 and Lemma 16, that each tree in $\mathcal{H}_k$ has linear rank-width $k+1$ and is minimal with respect to this property. Moreover, by Lemma 19 any tree of linear rank-width $k+1$ contains as a vertex-minor a tree in $\mathcal{H}_k$. So it is enough to prove that two trees in $\mathcal{H}_k$ are not locally equivalent. Bouchet has proved in [6] that two trees are locally equivalent if and only if they are isomorphic. Hence, since no two trees in $\mathcal{H}_k$ are isomorphic, we are done. □

6 Conclusion

We proved that linear rank-width and path-width coincide on forests, and we determined the linear clique-width of forests in terms of their path-width. Our proof method for the first result completely differs from our proof method for the second result. We believe that the second method can be adapted in order to obtain a shorter but non-constructive proof for the first result.

We obtained a linear time algorithm that computes the linear rank-width and the linear clique-width of any forest. Natural questions are: Is there a linear time algorithm that computes the linear rank-width (or linear clique-width) of distance-hereditary graphs or of series-parallel graphs? And, more generally, is there a polynomial time algorithm that computes the linear rank-width (or linear clique-width) of graphs of bounded rank-width?

We used the fact that linear rank-width and path-width coincide in forests to determine the set of minimal excluded acyclic vertex-minors for linear rank-width k. One can probably use the same technique to compute the set of *minimal excluded acyclic induced subgraphs* for linear rank-width and linear clique-width k. The complete set of minimal excluded vertex-minors for linear rank-width k is unknown and a next step could be to determine the set of distance-hereditary excluded vertex-minors for linear rank-width k (we know from [22] that the number is at least doubly exponential in k). In [20] it is proved that the size of the excluded vertex-minors for rank-width k is bounded by $(6^{k+1}-1)/5$, and similar results exist for tree-width and path-width [25]. Can we get a similar result for linear rank-width?

Clique-width and linear clique-width are not monotone with respect to the vertex-minor inclusion and are only known to be monotone with respect to the induced subgraph inclusion. Characterizing linear clique-width with respect to the induced subgraph inclusion seems to be a hard task and few results have been obtained [13,19]. Can we at least characterize the linear clique-width of co-graphs (which have clique-width at most 2) or in general of distance-hereditary graphs (which have clique-width at most 3) in order to identify the set of distance-hereditary excluded induced subgraphs for linear clique-width k?

Acknowledgement. We thank an anonymous referee for helpful comments.

References

1. Isolde Adler, Arthur M. Farley, and Andrzej Proskurowski. Obstructions for linear rankwidth at most 1. *J. Discrete Applied Mathematics*, 2013. To appear.
2. Daniel Bienstock and Paul D. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12(2):239–245, 1991.
3. Hans L. Bodlaender and Ton Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996.
4. Hans L. Bodlaender, Ton Kloks, and Dieter Kratsch. Treewidth and pathwidth of permutation graphs. *SIAM J. Discrete Math.*, 8(4):606–616, 1995.
5. Hans L. Bodlaender and Rolf H. Möhring. The pathwidth and treewidth of cographs. In John R. Gilbert and Rolf G. Karlsson, editors, *SWAT*, volume 447 of *Lecture Notes in Computer Science*, pages 301–309. Springer, 1990.
6. André Bouchet. Transforming trees by successive local complementations. *J. Graph Theory*, 12(2):195–207, 1988.
7. André Bouchet. Circle graph obstructions. *J. Comb. Theory, Ser. B*, 60(1):107–144, 1994.
8. Bruno Courcelle and Joost Engelfriet. *Graph Structure and Monadic Second-Order Logic, A Language-Theoretic Approach.* Cambridge University Press, 2012.
9. Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(1-3):77–114, 2000.
10. Jonathan A. Ellis, Ivan Hal Sudborough, and Jonathan S. Turner. The vertex separation and search number of a graph. *Inf. Comput.*, 113(1):50–79, 1994.
11. Michael R. Fellows, Frances A. Rosamond, Udi Rotics, and Stefan Szeider. Clique-width is np-complete. *SIAM J. Discrete Math.*, 23(2):909–939, 2009.
12. Robert Ganian. Thread graphs, linear rank-width and their algorithmic applications. In Costas S. Iliopoulos and William F. Smyth, editors, *IWOCA*, volume 6460 of *Lecture Notes in Computer Science*, pages 38–42. Springer, 2010.
13. Frank Gurski. Characterizations for co-graphs defined by restricted nlc-width or clique-width operations. *Discrete Mathematics*, 306(2):271–277, 2006.
14. Frank Gurski. Linear layouts measuring neighbourhoods in graphs. *Discrete Mathematics*, 306(15):1637–1650, 2006.
15. Frank Gurski and Egon Wanke. On the relationship between nlc-width and linear nlc-width. *Theor. Comput. Sci.*, 347(1-2):76–89, 2005.
16. Frank Gurski and Egon Wanke. The nlc-width and clique-width for powers of graphs of bounded tree-width. *Discrete Applied Mathematics*, 157(4):583–595, 2009.

17. Pinar Heggernes, Daniel Meister, and Charis Papadopoulos. A complete characterisation of the linear clique-width of path powers. In Jianer Chen and S. Barry Cooper, editors, *TAMC*, volume 5532 of *Lecture Notes in Computer Science*, pages 241–250. Springer, 2009.
18. Pinar Heggernes, Daniel Meister, and Charis Papadopoulos. Graphs of linear clique-width at most 3. *Theor. Comput. Sci.*, 412(39):5466–5486, 2011.
19. Pinar Heggernes, Daniel Meister, and Charis Papadopoulos. Characterising the linear clique-width of a class of graphs by forbidden induced subgraphs. *Discrete Applied Mathematics*, 160(6):888–901, 2012.
20. Sang il Oum. Rank-width and vertex-minors. *J. Comb. Theory, Ser. B*, 95(1):79–100, 2005.
21. Sang il Oum. Rank-width and well-quasi-ordering. *SIAM J. Discrete Math.*, 22(2):666–682, 2008.
22. Jisu Jeong, O joung Kwon, and Sang il Oum. Excluded vertex-minors for graphs of linear rank-width at most k. In Natacha Portier and Thomas Wilke, editors, *STACS*, volume 20 of *LIPIcs*, pages 221–232. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2013.
23. Lefteris M. Kirousis and Christos H. Papadimitriou. Searching and pebbling. *Theor. Comput. Sci.*, 47(3):205–218, 1986.
24. Ton Kloks and Hans L. Bodlaender. Approximating treewidth and pathwidth of some classes of perfect graphs. In Toshihide Ibaraki, Yasuyoshi Inagaki, Kazuo Iwama, Takao Nishizeki, and Masafumi Yamashita, editors, *ISAAC*, volume 650 of *Lecture Notes in Computer Science*, pages 116–125. Springer, 1992.
25. Jens Lagergren. Upper bounds on the size of obstructions and intertwines. *J. Comb. Theory, Ser. B*, 73(1):7–40, 1998.
26. Andrea S. LaPaugh. Recontamination does not help to search a graph. *J. ACM*, 40(2):224–245, 1993.
27. Vadim V. Lozin and Dieter Rautenbach. The relative clique-width of a graph. *J. Comb. Theory, Ser. B*, 97(5):846–858, 2007.
28. Nimrod Megiddo, S. Louis Hakimi, M. R. Garey, David S. Johnson, and Christos H. Papadimitriou. The complexity of searching a graph. *J. ACM*, 35(1):18–44, 1988.
29. Sang-il Oum and Paul D. Seymour. Approximating clique-width and branch-width. *J. Comb. Theory, Ser. B*, 96(4):514–528, 2006.
30. Petra Scheffler. Die Baumweite von Graphen als ein Maß für die Kompliziertheit algorithmischer Probleme. Akademie der Wissenschaften der DDR, Berlin, 1989. PhD thesis.
31. Karol Suchan and Ioan Todinca. Pathwidth of circular-arc graphs. In Andreas Brandstädt, Dieter Kratsch, and Haiko Müller, editors, *WG*, volume 4769 of *Lecture Notes in Computer Science*, pages 258–269. Springer, 2007.
32. Atsushi Takahashi, Shuichi Ueno, and Yoji Kajitani. Minimal acyclic forbidden minors for the family of graphs with bounded path-width. *Discrete Mathematics*, 127(1-3):293–304, 1994.