Comparing and Computing Parameters for Directed Graphs

Inaugural dissertation

for the attainment of the title of doctor
in the Faculty of Mathematics and Natural Sciences
at the Heinrich Heine University Düsseldorf

presented by

Carolin Rehs

from Düsseldorf

Düsseldorf, September 2021

from the institute for Computer Science
Faculty of Mathematics and Natural Sciences
at the Heinrich Heine University Düsseldorf

Published by permission of the
Faculty of Mathematics and Natural Sciences at
Heinrich Heine University Düsseldorf

Supervisor: PD Dr. Frank Gurski

Co-supervisor Prof. Dr. Egon Wanke

Date of the oral examination: 18/03/2022

Contents

1 Introduction

The field of undirected graph parameters is a very huge one and has been well-researched since the 1980s. A graph parameter, also called width measure, is a function that associates a positive integer with every graph. Probably the most considered undirected graph parameters in literature are path-width and tree-width [RS83, RS86a] as well as clique-width [CO00]. All of these parameters allow strong algorithmic results considering fixed-parameter-tractability. Especially for tree-width and clique-width there is an often cited Theorem by Courcelle, which states that all graph problems describable in monadic second-order logic on quantification over vertices and vertex sets (and additionally on quantification over edges and edge sets) are computable in polynomial time for bounded clique-width (tree-width).

This leads to the question, if there exist directed width measures that are as strong as the undirected ones. There have been numerous attempts to define directed width measures which admit as many algorithmical results as tree-width and path-width and there has been some well-known research to compare these parameters, see [GHK$^+$10] for a survey. Having studied this matter for some time, one comes to the conclusion that none of the known directed width parameters allows as strong results as the undirected ones. However, this does not mean that directed width parameters are not worth working on. Especially when not considered for all digraphs in general, but on special digraph classes, interesting results on these parameters remain possible.

In this work, several width measures on digraphs are considered. We investigate especially directed path-width (d-pw) and directed tree-width (d-tw), but particularly for tree-width there have been many different attempts to define a directed version. As the idea of undirected tree-width comes from Robertson and Seymour, it seems to be reasonable to consider mainly the paper by Johnson, Robertson, Seymour and Thomas in which they define not only a directed version of tree-width, but also directed path-width [JRST01b]. But even they themselves later published an addendum [JRST01a], in which they suggested to slightly change the definition.

We also consider the directed tree-width versions by Reed [Ree99], by Kreutzer and Ordyniak in Chapter 6 of the book "Quantitative Graph Theory" [DES14], by Kreutzer and Kwon in Chapter 9 of "Classes of Directed Graphs" [BJG18] and the idea of Courcelle and Olariu to use the underlying undirected graph. As an interesting fact we can see, that all these different definitions of directed tree-width but the last mentioned only differ by a constant factor, thus they are equivalent.

Other ideas for directed versions of the well-known tree-width that are studied in this work are DAG-width (dagw) [BDHK06, BDH+12, Obd06] and Kelly-width (kw) [HK08].

Directed Clique-width (d-cw) [CO00] and directed NLC-width (d-nlcw) [GWY16] and their linear versions are definable very similar to their undirected versions. Nevertheless they are interesting to consider, as they allow the strongest algorithmic results on directed graphs in general.

Further directed graph parameters are directed vertex separation number (d-vsn) [YC08], directed cut-width (d-cutw) [CFS12], directed neighbourhood-width (d-nw) [GR19a], directed linear rank-width (d-lrw) [GR19a], directed feedback vertex set number (fvs) [GKR19a], cycle rank (cr) [Egg63], DAG-depth (ddp) [GHK+09], and directed modular width [SW20].

A very new directed graph parameter which seems to allow some algorithmic aspects is directed branch-width [BMP20].

As there are such an amount of different parameters, it seems very likely to compare them. In the following, we first present linear, then non-linear directed graph parameters. Many linear width measures are comparable in general, whereas this is not possible for the non-linear ones. Therefore, we regard several directed graph classes and compare directed width parameters in this restricted area. On several graph classes, this leads to the computability of these graph parameters in polynomial or even linear time. We give algorithms for the computation of different graph parameters and relations between them on tree-like digraphs, directed co-graphs, their superclass directed distance-hereditary graphs, sequence digraphs and semicomplete graphs.

In Figure 1.1 the most important results on non-linear directed width measures in special graph classes are summarized.

Please note that in this work, many of our already published papers are used. An overview on my parts of these papers is given in the addendum.

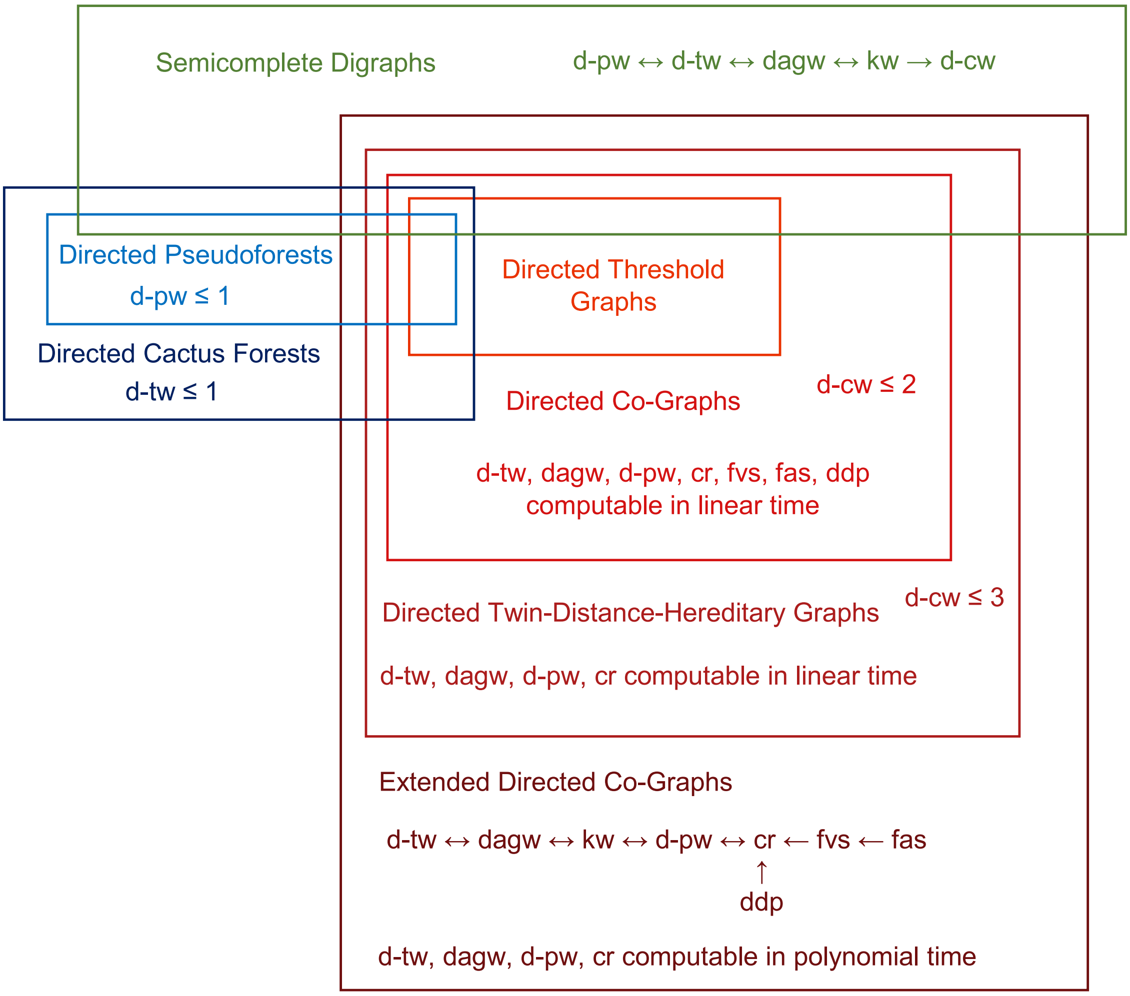


Figure 1.1: Relations between different graph parameters in special directed graph classes. The rectangles represent the sets of digraph classes and their relations to each other. Inside the rectangles are the results for the named graph classes. For a graph parameter α and a constant number c, $\alpha \leq c$ means that for every graph in this graph class, $\alpha(G)$ is lower or equal c. For two measures α and β, a directed edge from β to α indicates that there is some function f such that for every graph in this graph class it holds $\alpha(G) \leq f(\beta(G))$.

2 Undirected Graphs

First of all, we give some preliminary definitions and an introduction to the well-researched area of undirected width measures.

Please note that most of these preliminaries can be refound in several of our published papers.

Undirected graphs We work with finite undirected *graphs* $G = (V, E)$, where V is a finite set of *vertices* and $E \subseteq \{\{u, v\} \mid u, v \in V,\ u \neq v\}$ is a finite set of *edges*. For a vertex $v \in V$ we denote by $N_G(v)$ the set of all vertices which are adjacent to v in G, i.e. $N_G(v) = \{w \in V \mid \{v, w\} \in E\}$. Set $N_G(v)$ is called the set of all *neighbors* of v in G or *neighborhood* of v in G. The *degree* of a vertex $v \in V$, denoted by $\deg_G(x)$, is the number of neighbors of vertex v in G, i.e. $\deg_G(v) = |N_G(v)|$. The maximum vertex degree is $\Delta(G) = \max_{v \in V} \deg_G(v)$. A graph $G' = (V', E')$ is a *subgraph* of graph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. If every edge of E with both end vertices in V' is in E', we say that G' is an *induced subgraph* of digraph G and we write $G' = G[V']$. For some undirected graph $G = (V, E)$ its complement graph is defined by $\overline{G} = (V, \{\{u, v\} \mid \{u, v\} \notin E, u, v \in V, u \neq v\})$.

Special Undirected Graphs By $P_n = (\{v_1, \ldots, v_n\}, \{\{v_1, v_2\}, \ldots, \{v_{n-1}, v_n\}\})$, $n \geq 2$, we denote a path on n vertices and by $C_n = (\{v_1, \ldots, v_n\}, \{\{v_1, v_2\}, \ldots, \{v_{n-1}, v_n\}, \{v_n, v_1\}\})$, $n \geq 3$, we denote a cycle on n vertices. Further by $K_n = (\{v_1, \ldots, v_n\}, \{\{v_i, v_j\} \mid 1 \leq i < j \leq n\})$, $n \geq 1$, we denote a complete graph on n vertices and by $K_{n,m} = (\{v_1, \ldots, v_n, w_1, \ldots, w_m\}, \{\{v_i, w_j\} \mid 1 \leq i \leq n, 1 \leq j \leq m\})$ a complete bipartite graph on $n + m$ vertices.

By *hole*, *gem*, *house* and *domino* we denote the graphs given in Figure 2.1.

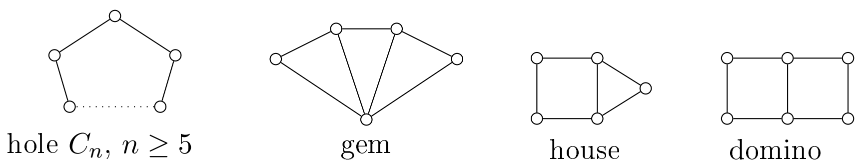


Figure 2.1: Hole, gem, house and domino graph

2.1 Graph Width Measures

2.1.1 Path-Width

As already mention in the introduction, the notion of path-width has been introduced by Robertson and Seymour in [RS83].

Definition 2.1.1 (Path-width)**.** [RS83] A path-decomposition of a graph $G = (V_G, E_G)$ is a sequence $(X_1, X_2, \dots, X_r)$ of subsets of V_G, such that the following three conditions hold true.

(pw-1) $X_1 \cup \ldots \cup X_r \ = \ V_G$.

(pw-2) For every edge $\{u, v\} \in E_G$ there is a set X_i, $1 \leq i \leq r$, such that $u, v \in X_i$.

(pw-3) For all i, j, ℓ with $1 \leq i < j < \ell \leq r$ it holds $X_i \cap X_\ell \subseteq X_j$.

The width of a path-decomposition $(X_1, \dots, X_r)$ is $\max_{1 \leq i \leq r} |X_i| - 1$. The path-width of a graph G, $\mathrm{pw}(G)$ for short, is the minimum width over all path-decompositions of G.

Determining whether the path-width of some given graph is at most some given value w is NP-complete [KF79] even for bipartite graphs, complements of bipartite graphs [ACP87], chordal graphs [Gus93], bipartite distance hereditary graphs [KBMK93], and planar graphs with maximum vertex degree 3 [MS88].

On the other hand, determining whether the path-width of some given graph is at most some given value w is polynomial for permutation graphs [BKK93], circular arc graphs [ST07], and co-graphs [BM93].

2.1.2 Tree-Width

One of the most famous tree structured graph classes are graphs of bounded tree-width. Though the concept of tree-width has already bin given in a work of Halin [Hal76], tree-width was defined in the 1980s by Robertson and Seymour in [RS86a] as follows.

Definition 2.1.2 (Tree-width)**.** [RS86a] A *tree decomposition* of a graph $G = (V_G, E_G)$ is a pair $(\mathcal{X}, T)$ where $T = (V_T, E_T)$ is a tree and $\mathcal{X} = \{X_u \mid \ u \in V_T\}$ is a family of subsets $X_u \subseteq V_G$, one for each node u of T, such that the following three conditions hold true.

(tw-1) $\cup_{u \in V_T} X_u = V_G$.

(tw-2) For every edge $\{v_1, v_2\} \in E_G$, there is some node $u \in V_T$ such that $v_1 \in X_u$ and $v_2 \in X_u$.

(tw-3) For every vertex $v \in V_G$ the subgraph of T induced by the nodes $u \in V_T$ with $v \in X_u$ is connected.

The *width* of a tree-decomposition $(\mathcal{X} = \{X_u \mid u \in V_T\},\ T = (V_T, E_T))$ is $\max_{u \in V_T} |X_u| - 1$. The *tree-width* of a graph G, $\text{tw}(G)$ for short, is the smallest integer k such that there is a tree-decomposition $(\mathcal{X}, T)$ for G of width k.

Determining whether the tree-width of some given graph is at most some given value w is NP-complete even for bipartite graphs and complements of bipartite graphs [ACP87].

2.1.3 Clique-Width

Before the naming of the clique-width established, its operations has been first considered by Courcelle, Engelfriet, and Rozenberg in [CER93]. The first mention of clique-width for labeled graphs is defined by Courcelle and Olariu in [CO00]. The following definition is taken from [GW00].

Definition 2.1.3 (Clique-width)**.** Let k be some positive integer. The class CW_k of labeled graphs is recursively defined as follows.

1. The single vertex graph $\bullet_a$ for some $a \in [k]$ is in CW_k.

2. Let $G = (V_G, E_G, \text{lab}_G) \in \text{CW}_k$ and $J = (V_J, E_J, \text{lab}_J) \in \text{CW}_k$ be two vertex-disjoint labeled graphs, then

$$G \oplus J := (V', E', \text{lab}')$$

   defined by $V' := V_G \cup V_J$, $E' := E_G \cup E_J$, and

$$\text{lab}'(u) := \begin{cases} \text{lab}_G(u) & \text{if } u \in V_G \\ \text{lab}_J(u) & \text{if } u \in V_J \end{cases}$$

   for every $u \in V'$ is in CW_k.

3. Let $a, b \in [k]$ be two distinct integers and $G = (V_G, E_G, \text{lab}_G) \in \text{CW}_k$ be a labeled graph, then

   (a) $\rho_{a \to b}(G) := (V_G, E_G, \text{lab}')$ defined by

$$\text{lab}'(u) := \begin{cases} \text{lab}_G(u) & \text{if } \text{lab}_G(u) \neq a \\ b & \text{if } \text{lab}_G(u) = a \end{cases}$$

   for every $u \in V_G$ is in CW_k and

   (b) $\eta_{a,b}(G) := (V_G, E', \text{lab}_G)$ defined by

$$E' := E_G \cup \{\{u, v\} \mid u, v \in V_G,\ u \neq v,\ \text{lab}_G(u) = a,\ \text{lab}_G(v) = b\}$$

   is in CW_k.

The *clique-width* of a labeled graph G, $\text{cw}(G)$ for short, is the least integer k such that $G \in \text{CW}_k$.

An expression X built with the operations $\bullet_a, \oplus, \rho_{a \to b}, \eta_{a,b}$ for integers $a, b \in [k]$ is called a *clique-width k-expression*. If integer k is known from the context or irrelevant for the discussion, then we sometimes use the simplified notion *expression* for the notion k-expression. The graph defined by expression X is denoted by $\text{val}(X)$. Every unlabeled graph $G = (V, E)$ is considered as the labeled graph (V, E, lab) where $\text{lab} : V \to [1]$.

It is NP-hard to compute clique-width when it is unbounded and unknown whether computable in polynomial time when bounded [FRRS09].

2.1.4 NLC-width

The notion of NLC-width (where NLC results from the *node label controlled* embedding mechanism for graph grammars) of labeled graphs is defined by Wanke in [Wan94] as follows.

Definition 2.1.4 (NLC-width)**.** Let k be some positive integer. The class NLC_k of labeled graphs is recursively defined as follows.

1. The single vertex graph $\bullet_a$ for some $a \in [k]$ is in NLC_k.

2. Let $G = (V_G, E_G, \text{lab}_G) \in \text{NLC}_k$ and $J = (V_J, E_J, \text{lab}_J) \in \text{NLC}_k$ be two vertex-disjoint labeled graphs and $S \subseteq [k]^2$ be a relation, then

$$G \times_S J := (V', E', \text{lab}')$$

defined by $V' := V_G \cup V_J$,

$$E' := E_G \cup E_J \cup \{\{u, v\} \mid u \in V_G,\ v \in V_J,\ (\text{lab}_G(u), \text{lab}_J(v)) \in S\},$$

and

$$\text{lab}'(u) \ := \ \begin{cases} \text{lab}_G(u) & \text{if } u \in V_G \\ \text{lab}_J(u) & \text{if } u \in V_J \end{cases}$$

for every $u \in V'$ is in NLC_k.

3. Let $G = (V_G, E_G, \text{lab}_G) \in \text{NLC}_k$ and $R : [k] \to [k]$ be a function, then

$$\circ_R(G) := (V_G, E_G, \text{lab}')$$

defined by

$$\text{lab}'(u) := R(\text{lab}_G(u))$$

for every $u \in V_G$ is in NLC_k.

The *NLC-width* of a labeled graph G, nlcw(G) for short, is the least integer k such that $G \in \mathrm{NLC}_k$.

An expression X built with the operations $\bullet_a, \times_S, \circ_R$ for $a \in [k]$, $S \subseteq [k]^2$, and $R : [k] \to [k]$ is called an *NLC-width k-expression*. If integer k is known from the context or irrelevant for the discussion, then we sometimes use the simplified notion *expression* for the notion k-expression. The graph defined by expression X is denoted by val(X). Every unlabeled graph $G = (V, E)$ is considered as the labeled graph (V, E, lab) where lab : $V \to [1]$.

Expression Trees Every NLC-width k-expression X has by its recursive definition a tree structure that is called the *NLC-width k-expression-tree* for X. This tree T is an ordered rooted tree whose leaves correspond to the vertices of graph val(X) and the inner nodes[1] correspond to the operations of X, see [GW00]. In the same way we define the clique-width k-expression-tree for every clique-width k-expression, see [EGW03]. If integer k is known from the context or irrelevant for the discussion, then we sometimes use the simplified notion *expression-tree* for the notion k-expression-tree. For some node u of expression-tree T, let $T(u)$ be the subtree of T rooted at u. Note that tree $T(u)$ is always an expression-tree. The expression $X(u)$ defined by $T(u)$ can simply be determined by traversing the tree $T(u)$ starting from the root, where the left children are visited first. $X(u)$ defines a (possibly) relabeled induced subgraph $G(u)$ of G. For an inner node v of some expression-tree T and a leaf u of $T(v)$ we define by lab($u, G(v)$) the label of that vertex of graph $G(v)$ that corresponds to u. A node u of T is called a *predecessor* of a node u' of T if u' is on a path from u to a leaf. A node u of T is called the *least common predecessor* of two nodes u_1 and u_2 if u is a predecessor of both nodes u_1, u_2, and no child of u is a predecessor of u_1, u_2.

Graph Parameters and Relations There is a very close relation between the clique-width and the NLC-width of a graph. We denote two expressions X_1 and X_2 as *equivalent*, if the unlabeled versions of val(X_1) and val(X_2) are isomorphic.

Theorem 2.1.5 ([Joh98]). *Every clique-width k-expression can be transformed into an equivalent NLC-width k-expression and every NLC-width k-expression can be transformed into an equivalent clique-width 2k-expression. Thus, for every graph G it holds*

$$nlcw(G) \leq cw(G) \leq 2 \cdot nlcw(G). \tag{2.1}$$

Computing the NLC-width of a graph is NP-hard [GW05a].

2.2 Graph Minors

Definition 2.2.1 (Edge contraction). Let $G = (V, E)$ be a graph with $e = \{u, v\} \in E$, $u \neq v$. The *contraction* of e leads to a new graph $G' = (V', E')$ with $V' =$

[1]The vertices in a tree can also be called *nodes*.

$V \backslash \{u, v\} \cup \{w\}$ with $w \notin V$ and $E' = \{\{a, b\} \mid a, b \in V \cap V',\ \{a, b\} \in E$ *or* $a = w, \{u, b\}$ *or* $\{v, b\} \in E$ *or* $b = w, \{a, u\}$ *or* $\{a, v\} \in E\}$.[2]

A graph minor of a graph $G = (V, E)$ is a graph $G' = (V', E')$, if G' can be obtained by forming subgraphs and edge contraction of G. Formally, we write $G' \preceq G$.

2.3 Graph Classes

2.3.1 Cactus trees and Pseudotrees

This subsection is taken from [GR19b].

Cactus trees are well-known in graph theory. The name "cactus" has been introduced by Harary and Uhlenbeck in 1953 [HU53]. The definition has slightly changed since then, whereas in the original definition cacti where requested to consist only of triangles, today's more common definition is as follows:

Definition 2.3.1 (Cactus tree)**.** A *cactus tree* is a connected graph $G = (V, E)$, where for any two cycles C_1 and C_2 it holds that they have at most one joint vertex.

The set of cactus graphs is a superset of the pseudotrees, which are again a superset of the well-known sunlet graphs.

Definition 2.3.2 (Pseudotree)**.** A *pseudotree* is a connected graph which contains at most one cycle.

It is possible to extend these definitions to forests, which means that they are not necessarily connected. A cactus forest is a graph where any two cycles have at most one joint vertex, and a pseudoforest is a graph where every connected component contains at most one cycle.

Whereas the set of cactus trees and the set of pseudotrees are not closed under the graph minor operation, as subgraphs could create unconnected graphs, cactus forests can be characterized by one forbidden graph minor, the diamond graph D_4 with four vertices, which is the K_4 with one edge less [EMC88]. This means, that every graph, which does not have D_4 as a graph minor, is a cactus forest. As pseudoforests are a subset of cactus forests, D_4 is also a forbidden minor for them, as well as the butterfly graph B_5. Every graph, which has neither D_4 nor B_5 as a graph minor, is a pseudoforest.

Cactus forests are of bounded tree-width and pseudoforests are even of bounded path-width.

2.3.2 Co-Graphs

Co-graphs have been introduced in the 1970s by a number of authors under different notations, such as hereditary Dacey graphs (HD graphs) in [Sum74], D^*-graphs in

[2] This means, in graph G' the edge e and its two incident vertices u and v are replaced by the vertex w and all other edges in G incident with u or v are adjacent with w in G'.

[Jun78], 2-parity graphs in [BU84], and complement reducible graphs (co-graphs) in [Ler71]. Co-graphs can be characterized as the set of graphs without an induced path with four vertices [CLSB81]. From an algorithmic point of view the following recursive definition is very useful.

This section is taken from [GKR21b]

Recursive Operations

Let $G_1 = (V_1, E_1), \ldots, G_k = (V_k, E_k)$ be k vertex-disjoint graphs.

- The *disjoint union* of $G_1, \ldots, G_k$, denoted by $G_1 \cup \ldots \cup G_k$, is the graph with vertex set $V_1 \cup \ldots \cup V_k$ and edge set $E_1 \cup \ldots \cup E_k$.
- The *join composition* of $G_1, \ldots, G_k$, denoted by $G_1 \times \ldots \times G_k$, is defined by their disjoint union plus all possible edges between vertices of G_i and G_j for all $1 \leq i, j \leq k$, $i \neq j$.

Co-graphs

Definition 2.3.3 (Co-graphs)**.** The class of *co-graphs* is recursively defined as follows.

(i) Every graph on a single vertex $(\{v\}, \emptyset)$, denoted by $\bullet$, is a *co-graph*.

(ii) If G_1, G_2 are vertex-disjoint co-graphs, then

  (a) the disjoint union $G_1 \cup G_2$ and

  (b) the join composition $G_1 \times G_2$ are *co-graphs*.

Please note that sometimes in literature, instead of the disjoint union and join composition of only G_1 and G_2 these operations on $G_1, \ldots, G_k$ are used in (ii). However, the defined graph class is the same, as $(G_1 \cup G_2) \cup G_3 = G_1 \cup G_2 \cup G_3$ and $(G_1 \times G_2) \times G_3 = G_1 \times G_2 \times G_3$.

By this definition every co-graph can be represented by a tree structure, denoted as *co-tree*. The leaves of the co-tree represent the vertices of the graph and the inner nodes of the co-tree correspond to the operations applied on the subexpressions defined by the subtrees. For every graph G one can decide in linear time, whether G is a co-graph and in the case of a positive answer construct a co-tree for G, see [HP05]. Using the co-tree a lot of hard problems have been shown to be solvable in polynomial time when restricted to co-graphs. Such problems are clique, independent set, partition into independent sets (chromatic number), partition into cliques, hamiltonian cycle, isomorphism [CLSB81].

2.3.3 Threshold Graphs

Threshold graphs were introduced by Chvátal and Hammer in the 1970s [CH77] as a graph class which allows to distinguish between independent and non-independent sets in a very simple way. Formally $G = (V, E)$ is a *threshold graph* if it can be constructed recursively by the following rules:

- The graph with only one vertex is a threshold graph.
- Let $G = (V, E)$ be a threshold graph. Then $G' = (V', E')$ with $V' = V \cup \{v\}$ for any vertex $v \notin V$ and $E' = E$ is a threshold graph. This operation is denoted as disjoint union $\cup$ or as adding an isolated vertex.
- Let $G = (V, E)$ be a threshold graph. Then $G' = (V', E')$ with $V' = V \cup \{v\}$ for any vertex $v \notin V$ and $E' = E \cup \{\{u, v\} \mid u \in V\}$ is a threshold graph. This operation is denoted as disjoint sum $\oplus$ or as adding a dominating vertex.

In [MP95] are presented some equivalent ways to define threshold graphs.
For every graph $G = (V, E)$, the following statements are equivalent:

1. G is a threshold graph.
2. There exist non-negative reals w_v, $v \in V$ and t such that for every $U \subseteq V$ it holds: $\sum_{v \in U} w_v \leq t$ iff U is an independent set of G.
3. G contains no C_4, no P_4 and no $2K_2$ as induced subgraph.
4. There exist non-negative reals w_v, $v \in V$ and T such that for every two vertices u, v it holds $w_u + w_v > T$ iff $\{u, v\} \in E$.
5. G is a split graph with vertex partition $V = S \cup K$ and the neighbourhoods of vertices of the independent set S are nested.
6. G and its edge complement graph $\overline{G}$ are trivially perfect.

2.3.4 Distance-Hereditary Graphs

Distance-hereditary graphs have been introduced by Howorka in 1977 [How77]. They are exactly the graphs which are distance-hereditary for their connected induced subgraphs, which means that if any two vertices u and v belong to an induced subgraph H of a graph G, then some shortest path between u and v in G has to be a subgraph of H.

But there are many equivalent definitions of distance-hereditary graphs, as shown in [BM86] and summarized in Theorem 10.1.1 of [BLS99]. Let G be a connected graph with distance function d. Then, the following conditions are equivalent:

1. G is distance-hereditary
2. For every two vertices u and v with $d(u, v) = 2$, there is no induced path between u and v of length greater than 2
3. The house, holes, domino, and gem are not induced subgraphs of G, i.e. G is HHDG-free

The Book [BLS99] contains one, the paper [BM86] even more characterizations, which are not used very often in graph theory and thus not necessary to mention here.

Further, distance-hereditary graphs can be defined recursively from a single vertex by the following three operations:

1. Add a pendant vertex, which is a vertex with only one edge to an existent vertex
2. Add a false twin, which is a vertex with the same neighbourhood as an existent vertex and no edge to this vertex
3. Add a true twin, which is a vertex with the same neighbourhood as an existent vertex and an edge to this vertex

As Co-Graphs are exactly the graphs obtained by the same rules without pendant vertices, distance-hereditary graphs are a superclass of co-graphs. Further, they are a subclass of perfect graphs.

3 Digraph (Width) Parameters

3.1 Preliminaries

We now give some elementary notations, which are mostly taken from [GR19a].

Let in general $[k] = \{1, \ldots, k\}$ be the set of all integers between 1 and k.

3.1.1 Directed graphs

A *directed graph* or *digraph* is a pair $G = (V, E)$, where V is a finite set of *vertices* and $E \subseteq \{(u, v) \mid u, v \in V,\ u \neq v\}$ is a finite set of ordered pairs of distinct[1] vertices called *arcs*. For a vertex $v \in V$, the sets $N_G^+(v) = \{u \in V \mid (v, u) \in E\}$ and $N_G^-(v) = \{u \in V \mid (u, v) \in E\}$ are called the *set of all successors* and the *set of all predecessors* of v in G. The sets $N_G^+(v) = \{u \in V \mid (v, u) \in E\}$ and $N_G^-(v) = \{u \in V \mid (u, v) \in E\}$ are called the *set of all out-neighbours* and the *set of all in-neighbours* of v. The *outdegree* of v, $\text{outdegree}_G(v)$ for short, is the number of out-neighbours of v and the *indegree* of v, $\text{indegree}_G(v)$ for short, is the number of in-neighbours of v in G. The *maximum out-degree* is $\Delta^+(G) = \max_{v \in V} \text{outdegree}_G(v)$ and the *maximum in-degree* is $\Delta^-(G) = \max_{v \in V} \text{indegree}_G(v)$. The *maximum vertex degree* is $\Delta(G) = \max_{v \in V} \text{outdegree}_G(v) + \text{indegree}_G(v)$. A digraph $G' = (V', E')$ is a *subdigraph* of digraph $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. If every arc of E with both end vertices in V' is in E', we say that G' is an *induced subdigraph* of digraph G and we write $G' = G[V']$.

For a set of digraphs $\mathcal{F}$ we denote by *$\mathcal{F}$-free digraphs* the set of all digraphs G such that no induced subdigraph of G is isomorphic to a member of $\mathcal{F}$. If $\mathcal{F} = \{F\}$, we write F-free instead of $\{F\}$-free. For some digraph class C we define $\text{Free}(C)$ as the set of all digraphs G such that no induced subdigraph of G is isomorphic to a member of C. This last two definitions can be used for undirected graphs as well.

For some digraph $G = (V, E)$ its *complement digraph* is defined by

$$\overline{G} = (V, \{(u, v) \mid (u, v) \notin E, u, v \in V, u \neq v\})$$

and its *converse digraph* is defined by

$$G^c = (V, \{(u, v) \mid (v, u) \in E, u, v \in V, u \neq v\}).$$

[1] Thus we do not consider directed graphs with loops.

Let $G = (V, E)$ be a digraph.

- G is *edgeless* if for all $u, v \in V$, $u \neq v$, none of the two pairs (u, v) and (v, u) belongs to E.
- G is a *tournament* if for all $u, v \in V$, $u \neq v$, exactly one of the two pairs (u, v) and (v, u) belongs to E.
- G is *semicomplete* if for all $u, v \in V$, $u \neq v$, at least one of the two pairs (u, v) and (v, u) belongs to E.
- G is *ℓ-semicomplete* if for all $u \in V$, there are at most ℓ vertices in V which are not neighbours (i.e. in- or out- neighbours) of u.
- G is *(bidirectional) complete* if for all $u, v \in V$, $u \neq v$, both of the two pairs (u, v) and (v, u) belong to E.

Omitting the directions For some given digraph $G = (V, E)$, we define its *underlying undirected graph* by ignoring the directions of the edges, i.e. $und(G) = (V, \{\{u, v\} \mid (u, v) \in E \text{ or } (v, u) \in E\})$.

Orientations There are several ways to define a digraph $G = (V, E)$ from an undirected graph $G_u = (V, E_u)$. If we replace every edge $\{u, v\} \in E_u$ by

- one of the arcs (u, v) and (v, u), we denote G as an *orientation* of G_u. Every digraph G which can be obtained by an orientation of some undirected graph G_u is called an *oriented graph*.
- one or both of the arcs (u, v) and (v, u), we denote G as a *biorientation* of G_u. Every digraph G which can be obtained by a biorientation of some undirected graph G_u is called a *bioriented graph*.
- both arcs (u, v) and (v, u), we denote G as a *complete biorientation* of G_u. Since in this case G is well defined by G_u we also denote it by $\overleftrightarrow{G_u}$. Every digraph G which can be obtained by a complete biorientation of some undirected graph G_u is called a *complete bioriented graph*.

Special directed graphs We now recall some special directed graphs. By $\overrightarrow{P_n} = (\{v_1, \ldots, v_n\}, \{(v_1, v_2), \ldots, (v_{n-1}, v_n)\})$, $n \geq 2$ we denote a directed path on n vertices and by $\overrightarrow{C_n} = (\{v_1, \ldots, v_n\}, \{(v_1, v_2), \ldots, (v_{n-1}, v_n), (v_n, v_1)\})$, $n \geq 2$ we denote a directed cycle on n vertices. The *k-power graph* G^k of a digraph G is a graph with the same vertex set as G. There is an arc (u, v) in G^k if and only if there is a directed path from u to v of length at most k in G. An *oriented forest (tree)* is the orientation of a forest (tree). A digraph is an *out-tree (in-tree)* if it is an oriented tree in which there is exactly one vertex of indegree (outdegree) zero. A *directed acyclic digraph (DAG for short)* is a digraph without any $\overrightarrow{C_n}$, $n \geq 2$ as subdigraph. Further let

$\overleftrightarrow{K_n} = (\{v_1, \ldots, v_n\}, \{(v_i, v_j) \mid 1 \leq i \neq j \leq n\})$ be a bidirectional complete digraph on n vertices.

3.1.2 Directed Graph Parameters

In order to classify graph parameters we call two graph parameters α and β *equivalent*, if there are two functions f_1 and f_2 such that for every digraph G the value $\alpha(G)$ can be upper bounded by $f_1(\beta(G))$ and the value $\beta(G)$ can be upper bounded by $f_2(\alpha(G))$.[2] If f_1 and f_2 are polynomials or linear functions, we call α and β *polynomially equivalent* or *linearly equivalent*, respectively.

3.2 Linear Width Parameters for Digraphs

In this section, we study directed graph parameters which are defined by the existence of an underlying path-structure for the input graph. Those parameters are called linear graph parameters, as a path-structure is, compared to a tree-structure, linear. The parameters of our interest are obtained by generalizing path-width [RS83], cut-width [AH73], linear clique-width [GW05b], linear NLC-width [GW05b], neighbourhood-width [Gur06b], and linear rank-width [Gan11] to directed graphs. With the exception of cut-width these parameters can be regarded as restrictions of the above mentioned parameters with underlying tree-structure to an underlying path-structure. The relation between these parameters corresponds to their tree-structural counterparts, since bounded path-width implies bounded linear NLC-width, linear clique-width, neighbourhood-width, and linear rank-width. Further the reverse direction is not true in general, see [Gur06b]. Such restrictions to underlying path-structures are often helpful to show results for the general parameters, see [FRRS09, FGL⁺18]. These linear parameters are also interesting from a structural point of view, e.g. in the research of special graph classes [Gan11, Gur06a, HMP11].

We now consider directed versions of the above mentioned linear paramters. Lifting them using an underlying tree-structure to directed graphs lead to the known directed tree-width [JRST01b], directed NLC-width [GWY16], directed clique-width [CO00], and directed rank-width [KR13].

One of the most famous examples for a directed graph parameter defined by the existence of an underlying path-structure is the directed path-width [RS83], which has been studied a lot [Bar06, Tam11, KKT15, KKK⁺16]. Further the cut-width for directed graphs was introduced by Chudnovsky et al. in [CFS12]. Regarding the usefulness of linear width parameters for undirected graphs we introduce the directed linear NLC-width, directed linear clique-width, directed neighbourhood-width, and directed linear rank-width. In contrast to the linear width measures for undirected graphs, for directed graphs their relations turn out to be more involved. Table 3.2

[2] Please note that two parameters are equivalent, if and only if the same families of digraphs have bounded width for them.

shows some classes of digraphs demonstrating various possible combinations of the listed width measures being bounded and unbounded.

	undirected		directed		DAG	CB	BS	OP	TT
cut-width	cutw	[AH73]	d-cutw	[CFS12]	0	∞	∞	0	0
path-width	pw	[RS83]	d-pw	Thomas et al.	0	∞	1	0	0
linear clique-width	lcw	[GW05b]	d-lcw	[GR19a]	∞	2	2	3	2
linear NLC-width	lnlcw	[GW05b]	d-lnlcw	[GR19a]	∞	1	1	3	1
neighbourhood-width	nw	[Gur06b]	d-nw	[GR19a]	∞	1	1	2	1
linear rank-width	lrw	[Gan11]	d-lrw	[GR19a]	∞	1	1	2	1

Table 3.1: Width measures and their values for directed acyclic digraphs (DAG), complete bioriented (CB) digraphs, bioriented stars (BS), oriented paths (OP), and transitive tournaments (TT).

For all these linear width parameters for directed graphs we compare the directed width of a digraph and the undirected width of its underlying undirected graph, which allow us to show the hardness of computing the considered linear width parameters for directed graphs.

We show that for general digraphs we have three sets of pairwise equivalent parameters, namely {d-cutw}, {d-pw}, and {d-nw, d-lnlcw, d-lcw, d-lrw}. For digraphs of bounded vertex degree this reduces to two sets {d-nw, d-lnlcw, d-lcw, d-lrw} and {d-cutw, d-pw} and for semicomplete digraphs of bounded vertex degree all these six graph parameters are pairwise equivalent. With the exception of directed rank-width, the same results are even shown for polynomially and linearly equivalence.

Please note this section is taken in huge parts from [GR19a].

3.2.1 Directed path-width

The path-width (pw) for undirected graphs was introduced in [RS83]. The notion of directed path-width was introduced by Reed, Seymour, and Thomas around 1995 (cf. [Bar06]) and relates to directed tree-width introduced by Johnson, Robertson, Seymour, and Thomas in [JRST01b]. Please note that there are some works which define the path-width of a digraph G in a different and not equivalent way by using the path-width of $und(G)$.

Definition 3.2.1 (directed path-width)**.** Let $G = (V, E)$ be a digraph. A *directed path-decomposition* of G is a sequence $(X_1, \dots, X_r)$ of subsets of V, called *bags*, such that the following three conditions hold true.

(dpw-1) $X_1 \cup \dots \cup X_r = V$,

(dpw-2) for each $(u, v) \in E$ there is a pair $i \leq j$ such that $u \in X_i$ and $v \in X_j$, and

(dpw-3) for all i, j, ℓ with $1 \leq i < j < \ell \leq r$ it holds $X_i \cap X_\ell \subseteq X_j$.

The *width* of a directed path-decomposition $\mathcal{X} = (X_1, \dots, X_r)$ is

$$\max_{1 \leq i \leq r} |X_i| - 1.$$

The *directed path-width* of G, d-pw(G) for short, is the smallest integer w such that there is a directed path-decomposition for G of width w.

A directed path-decomposition $(X_1, \dots, X_r)$ is *nice*, if $X_1 = \emptyset$, $X_r = \emptyset$, and $|X_i - X_{i-1}| + |X_{i-1} - X_i| = 1$ for every $2 \leq i \leq r$. If $X_i = X_{i-1} \cup \{v\}$, we denote X_i as an *introduce node* and if $X_{i-1} = X_i \cup \{v\}$, we denote X_i as a *forget node*. Every directed path-decomposition can be transformed into a nice directed path-decomposition in time $\mathcal{O}(|V|^2)$. By the definition of a path-decomposition, within a nice directed path-decomposition every graph vertex is introduced and forgotten exactly once. Thus every nice path-decomposition has $r = 2|V| + 1$ bags.

Example 3.2.2. In Figure 3.1 we show an illustration of a directed path-decomposition for a digraph G.

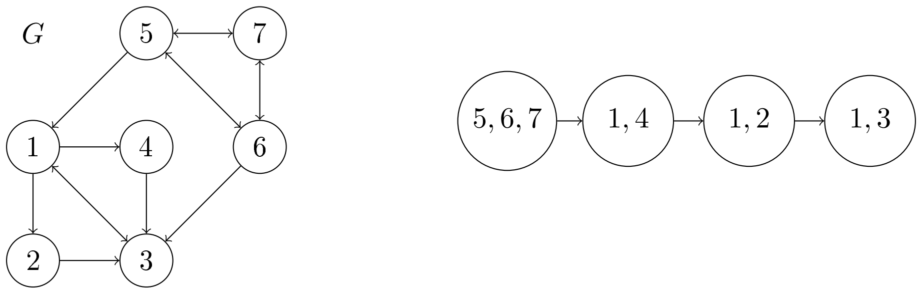


Figure 3.1: A digraph G (left) and a minimal path-decomposition for this graph (right).

Lemma 3.2.3. *Let G be some digraph, then d-pw$(G) \leq$ pw(und(G)).*

Lemma 3.2.4 (Lemma 1 of [Bar06])**.** *Let G be some complete bioriented digraph, then d-pw$(G) =$ pw(und(G)).*

The proof can be done straightforward since a for G of width k leads to a layout for $\overleftrightarrow{G}$ of width at most k and vice versa.

Determining whether the (undirected) path-width of some given (undirected) graph is at most some given value w is NP-complete [KF79] even for bipartite graphs, complements of bipartite graphs [ACP87], chordal graphs [Gus93], and planar graphs with maximum vertex degree 3 [MS88]. Lemma 3.2.4 implies that determining whether the directed path-width of some given digraph is at most some given value w is NP-complete even for digraphs whose underlying graphs lie in the mentioned classes. On the other hand, determining whether the (undirected) path-width of some given (undirected) graph is at most some given value w is polynomial for permutation graphs [BKK93], circular arc graphs [ST07], and co-graphs [BM93].

In the undirected cases, there are a number of results on path-width. But even in the directed case, there are already some algorithmic results for computing directed path-width. While undirected path-width can be solved by a fixed-parameter-algorithm [Bod96] (FPT-algorithm), the existence of such an algorithm for directed path-width is still open. The directed path-width of some directed graph $G = (V, E)$ can be computed in time $\mathcal{O}(|E|\cdot|V|^{2\text{d-pw}(G)}/(\text{d-pw}(G)-1)!)$ by [KKK+16] and in time $\mathcal{O}(\text{d-pw}(G)\cdot|E|\cdot|V|^{2\text{d-pw}(G)})$ by [Nag12]. This leads to XP-algorithms for directed path-width with respect to the standard parameter and implies that for each constant w, it is decidable in polynomial time whether a given digraph has directed path-width at most w. Further it is shown in [KKT15] how to decide whether the directed path-width of an ℓ-semicomplete digraph is at most w in time $(\ell + 2w + 1)^{2w} \cdot n^{\mathcal{O}(1)}$. Furthermore the directed path-width can be computed in time $3^{\tau(und(G))} \cdot |V|^{\mathcal{O}(1)}$, where $\tau(und(G))$ denotes the vertex cover number of the underlying undirected graph of G, by [Kob15].

The next lemma follows by the definition of converse digraphs and directed path-decompositions.

Lemma 3.2.5. *Let G be a digraph. Sequence $(X_1, \ldots, X_r)$ is a directed path-decomposition for G if and only if sequence $(X_r, \ldots, X_1)$ a directed path-decomposition of G^c.*

Lemma 3.2.6. *Let G be a digraph. Let G be some digraph, then d-pw(G) = d-pw(G^c).*

Example for digraphs of small directed path-width are given in Example 3.2.10, when considering the equivalent (cf. Lemma 3.5.15) notation of directed vertex separation number.

The directed path-width can change when taking the complement digraph, this is not possible if we restrict to tournaments.

Lemma 3.2.7. *[GKR21b] For every tournament G it holds d-pw(G) = d-pw$(\overline{G})$.*

Proof. By Lemma 3.2.6 we know that d-pw(G) = d-pw(G^c) and for tournaments G it holds d-pw(G^c) = d-pw$(\overline{G})$. □

Lemma 3.2.8. *Let G be a digraph, then the directed path-width of G is the maximum directed path-width of its strong components.*

Proof. Let G be a digraph, $AC(G)$ be the acyclic condensation of G, and $v_1, \ldots, v_c$ be a topological ordering of $AC(G)$, i.e. for every edge (v_i, v_j) in $AC(G)$ it holds $i < j$. Further let $V_1, \ldots, V_c$ be the vertex sets of its strong components ordered by the topological ordering. Then G can be obtained by $G = G[V_1] \ominus \ldots \ominus G[V_c]$. Since we have shown in [GR18] that d-pw$(G_1 \ominus G_2) = \max\{$d-pw$(G_1)$, d-pw$(G_2)\}$, the statement of the lemma follows.

□

3.2.2 Directed vertex separation number

The vertex separation number (vsn) for undirected graphs was introduced in [LT79].

In [YC08] the directed vertex separation number for a digraph $G = (V, E)$ has been introduced as follows.

Therefore we first need the definition of a *layout* of a graph $G = (V, E)$, which is a bijective function $\varphi : V \to \{1, \ldots, |V|\}$. For a graph G, we denote by $\Phi(G)$ the set of all layouts for G. Given a layout $\varphi \in \Phi(G)$ we define for $1 \leq i \leq |V|$ the vertex sets

$$L(i, \varphi, G) = \{u \in V \mid \varphi(u) \leq i\} \text{ and } R(i, \varphi, G) = \{u \in V \mid \varphi(u) > i\}.$$

The *reverse layout* φ^R, for $\varphi \in \Phi(G)$, is defined by $\varphi^R(u) = |V| - \varphi(u) + 1$, $u \in V$.

Definition 3.2.9 (directed vertex separation number, [YC08]). The *directed vertex separation number* of a digraph $G = (V, E)$ is defined as follows.

$$\text{d-vsn}(G) = \min_{\varphi \in \Phi(G)} \max_{1 \leq i \leq |V|} |\{u \in L(i, \varphi, G) \mid \exists v \in R(i, \varphi, G) : (v, u) \in E\}| \qquad (3.1)$$

For every optimal layout φ we obtain the same value when we consider the arcs forward in the reverse ordering φ^R. Thus we obtain an equivalent definition as follows (cf. [BJG09]).

$$\text{d-vsn}(G) = \min_{\varphi \in \Phi(G)} \max_{1 \leq i \leq |V|} |\{u \in R(i, \varphi, G) \mid \exists v \in L(i, \varphi, G) : (v, u) \in E\}| \qquad (3.2)$$

Since the converse digraph has the same path-width as its original graph, we obtain an equivalent definition, which will be useful later on.

$$\text{d-vsn}(G) = \min_{\varphi \in \Phi(G)} \max_{1 \leq i \leq |V|} |\{u \in L(i, \varphi, G) \mid \exists v \in R(i, \varphi, G) : (u, v) \in E\}| \qquad (3.3)$$

$$\text{d-vsn}(G) = \min_{\varphi \in \Phi(G)} \max_{1 \leq i \leq |V|} |\{u \in R(i, \varphi, G) \mid \exists v \in L(i, \varphi, G) : (u, v) \in A\}| \qquad (3.4)$$

Example 3.2.10 (directed vertex separation number). (1.) Every directed path $\overrightarrow{P_n}$ has directed vertex separation number 0.

(2.) The k-power graph $(\overrightarrow{P_n})^k$ of a directed path $\overrightarrow{P_n}$ has directed vertex separation number 0.

(3.) Every directed cycle $\overrightarrow{C_n}$ has directed vertex separation number 1. This can be shown by the layout $\varphi(v_i) = i$, $1 \leq i \leq n$.

(4.) The bidirectional complete digraph $\overleftrightarrow{K_3}$ and the complete biorientation of a star $K_{2,2,2}$ have directed vertex separation number 2.

(5.) Every bidirectional complete digraph $\overleftrightarrow{K_n}$ has directed vertex separation number $n - 1$.

3.2.3 Directed cut-width

The cut-width (cutw) of undirected graphs has been introduced in [AH73]. The cut-width of digraphs was introduced by Chudnovsky, Fradkin, and Seymour in [CFS12]. The directed cut-width of some digraph $G = (V, E)$ is defined by an ordering of vertices similar to undirected cut-width, with the exception that only arcs directed forward in the ordering contribute to the width of a cut.

Definition 3.2.11 (directed cut-width, [CFS12])**.** The *directed cut-width* of digraph $G = (V, E)$ is

$$\text{d-cutw}(G) = \min_{\varphi \in \Phi(G)} \max_{1 \leq i \leq |V|} |(u, v) \in E \mid u \in L(i, \varphi, G), v \in R(i, \varphi, G)\}|. \quad (3.5)$$

For every optimal layout φ we obtain the same value when we consider the arcs backwards in the reverse ordering φ^R. Thus we obtain an equivalent definition, which will be useful later on.

$$\text{d-cutw}(G) = \min_{\varphi \in \Phi(G)} \max_{1 \leq i \leq |V|} |(v, u) \in E \mid u \in L(i, \varphi, G), v \in R(i, \varphi, G)\}| \quad (3.6)$$

Subexponential parameterized algorithms for computing the directed cut-width of semicomplete digraphs are given in [FP13b].

Example 3.2.12 (directed cut-width). (1.) Every directed path $\overrightarrow{P_n}$ has directed cut-width 0. This can be shown by the layout $\varphi(v_i) = n - i + 1$, $1 \leq i \leq n$.

(2.) The k-power graph $(\overrightarrow{P_n})^k$ of a directed path $\overrightarrow{P_n}$ has directed directed cut-width 0.

(3.) Every directed cycle $\overrightarrow{C_n}$ has directed cut-width 1.

(4.) The bidirectional complete digraph $\overleftrightarrow{K_3}$ has directed cut-width 2.

(5.) Every bidirectional complete digraph $\overleftrightarrow{K_n}$ has directed cut-width $\lfloor \frac{n}{2} \rfloor \cdot \lceil \frac{n}{2} \rceil$.

3.2.4 Directed linear NLC-width

The linear NLC-width (lnlcw) for undirected graphs was introduced in [GW05b] as a parameter by restricting the NLC-width, defined in [Wan94], to an underlying path-structure. In [GR19a] we introduce the corresponding parameter for directed graphs by a modification of the edge inserting operation $\times_S$ of the linear NLC-width, which also leads to a restriction of directed NLC-width [GWY16].

Definition 3.2.13 (directed linear NLC-width)**.** The class of directed linear NLC-width at most k, $d - \text{lNLC}_k$ for short, is recursively defined as follows

1. Creation of a new vertex with label a, denoted by $\bullet_a$ for some $a \in [k]$ is in $d - \text{NLC}_k$.

2. Disjoint union of a labeled digraphs $G = (V, E, \text{lab}) \in d - \text{lNLC}_k$ and a single vertex $v \notin V$ labeled by a with two relations $\overrightarrow{S}, \overleftarrow{S} \in [k]^2$, denoted by

$$G \otimes_{(\overrightarrow{S}, \overleftarrow{S})} \bullet_a := (V', E', \text{lab}'),$$

where $V' := V \cup \{v\}$,

$$E' := E_G \cup \{(u, v) \mid u \in V,\ (\text{lab}(u), a) \in \overrightarrow{S}\} \cup \{(v, u) \mid u \in V,\ (\text{lab}(u), a) \in \overleftarrow{S}\},$$

and

$$\text{lab}'(u) := \begin{cases} \text{lab}(u) & \text{if } u \in V_G \\ a & \text{if } u = v \end{cases}$$

for every $u \in V'$ is in $d - \text{lNLC}_k$.

3. Change every label a in $G = (V, E, \text{lab})$ into label $R(a)$ by some function $R : [k] \to [k]$ denoted by $\circ_R := (V, E, \text{lab}')$ with

$$\text{lab}'(u) := R(\text{lab}(u))$$

for every $u \in V_G$ is in $d - \text{lNLC}_k$.

The *directed linear NLC-width* of an unlabeled digraph $G = (V, E)$ is the smallest integer k, such that there is a mapping lab : $V \to [k]$ such that the labeled digraph (V, E, lab) is in $d - \text{lNLC}_k$.

An expression X built with the operations defined above is called a *directed linear NLC-width k-expression.* Note that every expression defines a layout by the order in which the vertices are inserted in the corresponding digraph.

Every such expression has by its recursive definition a tree structure which we call the *directed linear NLC-width expression tree.*

Example 3.2.14 (directed linear NLC-width). (1.) Every bidirectional complete digraph $\overleftrightarrow{K_n}$ has directed linear NLC-width 1.

(2.) The directed paths $\overrightarrow{P_3}$ and $\overrightarrow{P_4}$ have directed linear NLC-width 2.

(3.) Every directed path $\overrightarrow{P_n}$ has directed linear NLC-width at most 3.

(4.) Every directed cycle $\overrightarrow{C_n}$ has directed linear NLC-width at most 4.

(5.) Every k-power graph $(\overrightarrow{P_n})^k$ of a directed path $\overrightarrow{P_n}$ has directed linear NLC-width at most $k + 2$.

(6.) Every complete biorientation of a grid $\overleftrightarrow{G_n}$, $n \geq 3$, has directed linear NLC-width at least n and at most $n + 2$, see [GR00, Gur08].

3.2.5 Directed linear clique-width

The linear clique-width (lcw) for undirected graphs was introduced in [GW05b] as a parameter by restricting the clique-width, defined in [CO00], to an underlying path-structure. Next we introduce the corresponding parameter for directed graphs by a modification of the edge inserting operation of the linear clique-width, which also leads to a restriction for directed clique-width [CO00].

Definition 3.2.15 (directed linear clique-width)**.** The class of directed linear clique-width at most k, $d-\mathrm{lCW}_k$ for short, is recursively defined as follows:

1. Creation of a new vertex with label a, denoted by $\bullet_a$, for some $a \in [k]$ is in $d-\mathrm{lCW}_k$.

2. Disjoint union of a labeled digraph $G = (V, E, \mathrm{lab}) \in d-\mathrm{lCW}_k$ and a single vertex labeled by a, denoted by $G \oplus \bullet_a$ where

$$G \oplus \bullet_a = (V', E, \mathrm{lab}')$$

   defined by $V' := V \cup \{v\}$ with $v \notin V$ and

$$\mathrm{lab}'(u) \;:=\; \begin{cases} \mathrm{lab}_G(u) & \text{if } u \in V_G \\ a & \text{if } u = v \end{cases}$$

   for every $u \in V'$ is in $d-\mathrm{lCW}_k$.

3. For $G = (V, E, \mathrm{lab}) \in d-\mathrm{lCW}_k$, inserting an arc from every vertex with label a to every vertex with label b, where $a, b \in [k]$, $a \neq b$, denoted by $\alpha_{a,b} := (V, E', \mathrm{lab})$ with

$$E' := E \cup \{(u, v) \mid u, v \in V,\ u \neq v,\ \mathrm{lab}(u) = a,\ \mathrm{lab}(v) = b\}$$

   is in $d-\mathrm{lCW}_k$.

4. For $G = (V, E, \mathrm{lab}) \in d-\mathrm{lCW}_k$, change label a into label b, denoted by $\rho_{a\to b} = (V, E, \mathrm{lab}')$ with

$$\mathrm{lab}'(u) \;:=\; \begin{cases} \mathrm{lab}(u) & \text{if } \mathrm{lab}(u) \neq a \\ b & \text{if } \mathrm{lab}(u) = a \end{cases}$$

   for every $u \in V_G$ is in $d-\mathrm{lCW}_k$.

The *directed linear clique-width* of an unlabeled digraph $G = (V, E)$ is the smallest integer k, such that there is a mapping $\mathrm{lab} : V \to [k]$ such that the labeled digraph (V, E, lab) is in $d-\mathrm{lCW}_k$.

An expression X built with the operations defined above is called a *directed linear clique-width k-expression.* Note that every expression defines a layout by the order in which the vertices are inserted in the corresponding digraph. Every such expression has by its recursive definition a tree structure which we call the *directed linear clique-width expression tree.*

Example 3.2.16 (directed linear clique-width). (1.) Every edgeless digraph has directed linear clique-width 1.

(2.) Every bidirectional complete digraph $\overleftrightarrow{K_n}$ has directed linear clique-width 2.

(3.) Every directed path $\overrightarrow{P_n}$ has directed linear clique-width at most 3.

(4.) Every directed cycle $\overrightarrow{C_n}$ has directed linear clique-width at most 4.

(5.) Every k-power graph $(\overrightarrow{P_n})^k$ of a directed path $\overrightarrow{P_n}$ has directed linear clique-width at most $k+2$. For $n \geq k(k+1)+2$ the given bound on the directed linear clique-width is even exact by Corollary 3.5.4.

(6.) Every complete biorientation of a grid $\overleftrightarrow{G_n}$, $n \geq 3$, has directed linear clique-width at least n and at most $n+2$, see [GR00, Gur08].

3.2.6 Directed neighbourhood-width

The neighbourhood-width (nw) for undirected graphs was introduced in [Gur06b]. It differs from linear NLC-width and linear clique-width at most by one but it is independent of vertex labels.

Let $G = (V, E)$ be a digraph and $U, W \subseteq V$ two disjoint vertex sets. The set of all out-neighbours of u into set W and the set of all in-neighbours of u into set W are defined by $N^+_W(u) = \{v \in W \mid (u, v) \in E\}$ and $N^-_W(u) = \{v \in W \mid (v, u) \in E\}$. The *directed neighbourhood* of vertex u into set W is defined by $N_W(u) = (N^+_W(u), N^-_W(u))$ and the set of all directed neighbourhoods of the vertices of set U into set W is $N(U, W) = \{N_W(u) \mid u \in U\}$. For some layout $\varphi \in \Phi(G)$ we define d-nw$(\varphi, G) = \max_{1\leq i\leq |V|} |N(L(i, \varphi, G), R(i, \varphi, G))|$.

Definition 3.2.17 (directed neighbourhood-width)**.** The *directed neighbourhood-width* of digraph G is

$$\text{d-nw}(G) = \min_{\varphi\in\Phi(G)} \text{d-nw}(\varphi, G).$$

Example 3.2.18 (directed neighbourhood-width). (1.) Every bidirectional complete digraph $\overleftrightarrow{K_n}$ has directed neighbourhood-width 1.

(2.) Every directed path $\overrightarrow{P_n}$ has directed neighbourhood-width at most 2.

(3.) Every directed cycle $\overrightarrow{C_n}$ has directed neighbourhood-width at most 3.

(4.) Every k-power graph $(\overrightarrow{P_n})^k$ of a directed path $\overrightarrow{P_n}$ has directed neighbourhood-width at most $k+1$. For $n \geq k(k+1)+2$ the given bound on the directed neighbourhood-width is even exact by Corollary 3.5.4.

(5.) Every complete biorientation of a grid $\overleftrightarrow{G_n}$, $n \geq 3$, has directed neighbourhood-width at least n and at most $n+1$, see [GR00, Gur08].

3.2.7 Directed linear rank-width

The rank-width for directed graphs was introduced by Kanté in [KR13]. In [Gan11] the linear rank-width (lrw) for undirected graphs was introduced by restricting the tree-structure of a rank decomposition to caterpillars, which is also possible for the directed case as follows.

Let $G = (V, E)$ a digraph and $V_1, V_2 \subset V$ be a disjoint partition of the vertex set of G. Further let $M_{V_1}^{V_2} = (m_{ij})$ be the adjacent matrix defined over the four-element field GF(4) for partition $V_1 \cup V_2$, i.e.

$$m_{ij} = \begin{cases} 0 & \text{if } (v_i, v_j) \notin E \text{ and } (v_j, v_i) \notin E \\ \eth & \text{if } (v_i, v_j) \in E \text{ and } (v_j, v_i) \notin E \\ \eth^2 & \text{if } (v_i, v_j) \notin E \text{ and } (v_j, v_i) \in E \\ 1 & \text{if } (v_i, v_j) \in E \text{ and } (v_j, v_i) \in E \end{cases}$$

In GF(4) we have four elements $\{0, 1, \eth, \eth^2\}$ with the properties $1 + \eth + \eth^2 = 0$ and $\eth^3 = 1$.

Definition 3.2.19 (directed linear rank-width)**.** A *directed linear rank decomposition* of digraph $G = (V, E)$ is a pair (T, f), where T is a caterpillar (i.e. a path with pendant vertices) and f is a bijection between V and the leaves of T. Each edge e of T divides the vertex set of G by f into two disjoint sets A_e, B_e. For an edge e in T we define the width of e as $\text{rg}^{(4)}(M_{A_e}^{B_e})$, i.e. the matrix rank of M. The *width* of a directed linear rank decomposition (T, f) is the maximal width of all edges in T. The *directed linear rank-width* of a digraph G, d-lrw(G) for short, is the minimum width of all directed linear rank decompositions for G.

Example 3.2.20 (directed linear rank-width). (1.) Every bidirectional complete digraph $\overleftrightarrow{K_n}$ and every directed path $\overrightarrow{P_n}$ has directed linear rank-width 1.

(2.) Every directed cycle $\overrightarrow{C_n}$ has directed linear rank-width at most 2.

(3.) Every complete biorientation of a grid $\overleftrightarrow{G_n}$, $n \geq 3$, has directed linear rank-width at least $\lceil \frac{2n}{3} \rceil$ and at most $n + 1$, see [HOSG08, Gur08].

3.3 Non-Linear Width Parameters for Digraphs

We now come to consider directed width measures with a non-linear underlying structure. As can be seen lateron, many of these parameters are not comparable in general. Therefore, we give only definitions and some general minor results in this section and reconsider non-linear directed graph parameters on special graph classes later in this work.

Please note that huge parts of this section are taken from [GKR21b].

3.3.1 Directed tree-width

Tree-width is a well-known parameter for undirected graphs. It has been defined independently by several researchers since 1972. Some of the most famous works are by Courcelle, Bodlaender, Robertson and Seymour.

Since then, there have been many attempts to define a directed version of tree-width. The most simple one has been defined by Courcelle, setting for the directed tree-width of a directed graph the tree-width of its underlying undirected graph.

In [JRST01b], Johnson, Robertson, Seymour and Thomas defined an "aboreal tree-decomposition" for directed graphs, which seems to be the most famous directed tree-width definition.

This definition of directed tree-width differs very much from the undirected version, but several important properties for undirected tree-width on undirected graphs remain fulfilled using this directed aboreal tree-width on directed graphs.

In a later published addendum to this paper [JRST01a], the authors correct the using of normality by a new definition of regularity, which strongly differs from normality. However, though the definitions lead to very different versions of directed tree-decompositions, the obtained directed tree-widths only differ by a constant factor.

Further, in this addendum an aboreal predecomposition is defined, which allows empty sets for the sets $W \subseteq \mathcal{W}$. There is also presented a linkage between this predecomposition and directed tree-width, with further properties as strongly connected components.

These ideas seemed to lead to the definition of directed tree-width by Kreutzer and Ordyniak in Chapter 6 of the book "Quantitative Graph Theory" [DES14]. They do not work with the terms of normality or regularity, but with strong components. In their definition, for all $e = (s, t) \in V$ the set $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}}$ has to be a strong component of $G - X_e$. Further, the sets $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}}$ and X_e have to be disjoint. It is not mentioned whether the sets $W \in \mathcal{W}$ have to be non-empty or not. In this book the authors remark that this definition differs from the definition in [JRST01b], but only in a constant factor.

In Chapter 9 of "Classes of Directed Graphs" [BJG18], Kreutzer and Kwon define directed tree-width using so-called strong guards. This definition looks very similar to the original one in [JRST01b], but a huge difference is that, as possibly in the addendum [JRST01a], the sets $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}}$ and X_e do not need to be disjoint.

A different definition of directed tree-width has been given by Reed in [Ree99]. But though the definition seems not very similar, it is strongly connected to the one of Johnson et al.

Though directed tree-width has been investigated a lot in the last years and is surely one of the most famous directed graph parameters [Bod98, Adl07, GR19b, GR18, Wie20], not everyone seems to be aware that it is not at all clear what is meant by "the" directed tree-width. Important results are only proven for one of the definitions, but not for all of them. But by comparing them, one comes to the conclusion, that most of these definition are equivalent, as they only differ by a constant factor. Therefore, many results are applicable to other definitions.

Underlying directed tree-width (u-d-tw)

In [CO00], Courcelle and Olariu use the definition of undirected tree-width also for directed graphs, which is possible for Definition 2.1.2, as adjacency is defined on undirected as well as on directed graphs. This leads to the equivalent definition:

Definition 3.3.1. Let G be a directed graph. Then $\text{u-d-tw}(G) = \text{tw}(und(G))$.

For this definition it is possible to bound directed clique-width by directed tree-width:

Lemma 3.3.2 ([CO00])**.** *Let G be a directed graph. Then $d\text{-}cw(G) \leq 2^{u\text{-}d\text{-}tw(G)+1}+1$.*

Bounding u-d-tw by directed clique-width is not possible on the set of all graphs.

Aboreal Directed Tree-Width

We now give the definition of Johnson et al. [JRST01b], which has been named aboreal directed tree-width. We will modify it a little bit, by allowing empty sets, as will be seen in the definition. But this does only changes to the original by a constant factor. Please note, that this is the definition, with which we will mostly work in the following sections. We therefore simply name it by directed tree-width, d-tw for short.

A *directed walk* in digraph $G = (V, E)$ is an alternating sequence $W = (u_1, e_1, u_2, e_2, u_3, \ldots, e_{k-1}, u_k)$ of vertices $v_i \in V$, $1 \leq i \leq k$, and edges $e_i \in E$, $1 \leq i \leq k-1$, such that $e_i = (u_i, u_{i+1})$, $1 \leq i \leq k-1$. If the vertices of the directed walk W are distinct, then W is a *directed path*.

An *acyclic* digraph (*DAG* for short) is a digraph without any cycles as subdigraph. An *out-tree* or *aboreal tree* is an orientation of a tree with a distinguished root such that all arcs are directed away from the root. For two vertices u, v of an out-tree T the notation $u \leq v$ means that there is a directed path on ≥ 0 arcs from u to v and $u < v$ means that there is a directed path on ≥ 1 arcs from u to v.

Let $G = (V, E)$ be some digraph and $Z \subseteq V$. The digraph $G[V - Z]$ which is obtained from G by deleting Z will be denoted by $G - Z$. A vertex set $S \subseteq V \setminus Z$ is Z*-normal* if there is no directed walk in $G - Z$ with first and last vertices in S that uses a vertex of $G - (Z \cup S)$. That is, a set $S \subseteq V$ is Z-normal, if every directed walk which leaves and again enters S in $G - Z$ must contain only vertices from $Z \cup S$. Or, a set $S \subseteq V$ is Z-normal, if every directed walk which leaves and again enters S must contain a vertex from Z see [BJG09].

In an aboreal tree, we define the set $W_{>v} = \bigcup_{\tilde{v}>r} W_{\tilde{v}}$ as the union of all sets $W_{\tilde{v}}$ of all (indirect) successors $\tilde{v}$ of v. As $W_{\geq v} = \bigcup_{\tilde{v}\geq v} W_{\tilde{v}}$ we define the union of the sets $W_{\tilde{v}}$ of all (indirect) successors $\tilde{v}$ of v including W_v.

Definition 3.3.3 (directed tree-width, [JRST01b]). A *(arboreal) tree-decomposition* of a digraph $G = (V_G, E_G)$ is a triple $(T, \mathcal{X}, \mathcal{W})$. Here $T = (V_T, E_T)$ is an out-tree, $\mathcal{X} = \{X_e \mid e \in E_T\}$ and $\mathcal{W} = \{W_r \mid r \in V_T\}$ are sets of subsets of V_G, such that the following two conditions hold true.

(dtw-1) $\mathcal{W} = \{W_r \mid r \in V_T\}$ is a partition of V_G into possibly empty subsets.[3]

(dtw-2) For every $(u, v) \in E_T$ the set $\bigcup\{W_r \mid r \in V_T, v \leq r\}$ is $X_{(u,v)}$-normal.

The *width* of a (aboreal) tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ is

$$\max_{r \in V_T} |W_r \cup \bigcup_{e \sim r} X_e| - 1.$$

Here $e \sim r$ means that r is one of the two vertices of arc e. The *directed tree-width* of G, d-tw(G) for short, is the smallest integer k such that there is a (arboreal) tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of G of width k.

In the following, by directed tree-decomposition is meant an aboreal directed tree-decomposition.

Example 3.3.4. In Figure 3.2 we show an illustration of an aboreal tree-decomposition for a digraph G.

Every DAG has directed tree-width 0. Furthermore, several types of tree-like digraphs have directed tree-width 1 [GR19b].

Remark 3.3.5 (Z-normality and Z-regularity). Please note that our definition seems to differ of Z-normality from the following definition in [JRST01b] where S and Z are disjoint. But as $S \subseteq V \setminus Z$, the sets S and Z are disjoint by definition.

Note that this means, that we use exactly the same definition for normality as in [JRST01b].

In their addendum [JRST01a], Johnson et al. suggested to use a definition, where Z and S are not necessarily disjoint, which means that $S \subseteq V$. They name this different definition Z-normality, which is formally defined by: Let $S, Z \in V$. The set S is Z-regular, if every directed walk which leaves and again enters S must contain a vertex from Z

The usage of regularity instead of normality leads to a different definition of aboreal directed tree-width. However, this new width only differs in a constant factor and all important theorems remain true.

We now give some properties about aboreal directed tree-width.

Lemma 3.3.6 ([JRST01b]). *Let G be some digraph, then d-tw$(G) \leq$ tw(und(G)).*

Lemma 3.3.7 ([JRST01b]). *Let G be some complete bioriented digraph, then d-tw(G) = tw(und(G)).*

[3] A remarkable difference to the undirected tree-width [RS86a] is that the sets W_r have to be disjoint.

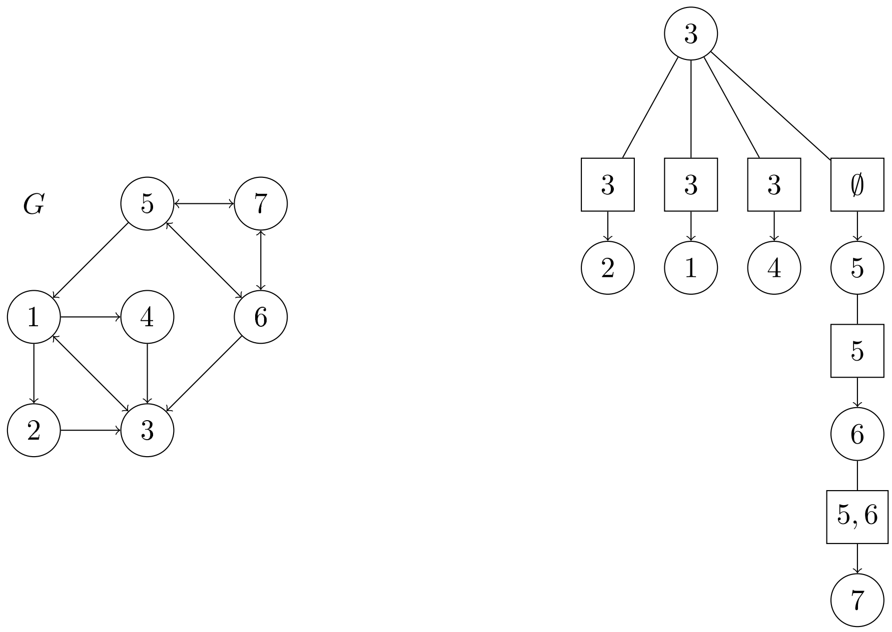


Figure 3.2: On the left a digraph G which is used in Examples 3.3.4, 3.3.28, 3.3.32 to illustrate decompositions of width measures. On the right an aboreal tree-decomposition of width 1 for digraph G. In the aboreal tree-decomposition, the X_e set is represented as a square on the respective edge e, while the W sets are represented as round vertices of the out-tree.

Determining whether the (undirected) tree-width of some given (undirected) graph is at most some given value w is NP-complete even for bipartite graphs and complements of bipartite graphs [ACP87]. Lemma 3.3.7 implies that determining whether the directed tree-width of some given digraph is at most some given value w is NP-complete even for digraphs whose underlying graphs lie in the mentioned classes.

The results of [JRST01b] lead to an XP-algorithm for directed tree-width w.r.t. the standard parameter which implies that for each constant w, it is decidable in polynomial time whether a given digraph has directed tree-width at most w.

Lemma 3.3.8 ([JRST01b]). *Let G be some digraph and H be an induced subdigraph of G, then d-tw$(H) \leq$ d-tw(G).*

Lemma 3.3.9. *Let G be a digraph of directed tree-width at most k. Then, there is a directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$, $T = (V_T, E_T)$, of width at most k for G such that $|W_r| \leq 1$ for every $r \in V_T$.*

Proof. Let $G = (V, E)$ be a digraph and $(T, \mathcal{X}, \mathcal{W})$, $T = (V_T, E_T)$, be a directed tree-decomposition of G. For every $r \in V_T$ such that $|W_r| \leq 1$ the statement of the lemma is fulfilled. Let $r \in V_T$ such that $W_r = \{v_1, \ldots, v_m\}$ for some $m > 1$. Further, let p be the predecessor of r in T and $s_1, \ldots, s_\ell$ be the successors of r in T. Let $(T', \mathcal{X}', \mathcal{W}')$ be defined by the following modifications of $(T, \mathcal{X}, \mathcal{W})$: We replace vertex r in T by the directed path $P(r) = (\{r_1, \ldots, r_m\}, \{(r_1, r_2), \ldots, (r_{m-1}, r_m)\})$ and replace arc (p, r) by (p, r_1) and the ℓ arcs (r, s_j), $1 \leq j \leq \ell$, by the ℓ arcs (r_m, s_j), $1 \leq j \leq \ell$ in T'. We define the sets $W'_{r_j} = \{v_j\}$ for $1 \leq j \leq m$. Further, we define the sets $X'_{(p,r_1)} = X_{(p,r)}$, $X'_{(r_m,s_j)} = X_{(r,s_j)}$, $1 \leq j \leq \ell$, and $X'_{(r_j,r_{j+1})} = X_{(p,r)} \cup \{r_1, \ldots, r_j\}$, $1 \leq j \leq m-1$.

By our definition, $\mathcal{W}'$ leads to a partition of V. The normality holds for the arcs of T' as follows. First, we consider the arcs (r_{i-1}, r_i), $1 < i \leq m$, which we inserted for sets W_r of size $m > 1$. The set $W'_{\geq r_i}$ is $X'_{(r_{i-1},r_i)}$-normal since $W_{\geq r}$ is $X_{(p,r)}$-normal and $X'_{(r_{i-1},r_i)} = X_{(p,r)} \cup \{r_1, \ldots, r_{i-1}\}$. Further, the property is fulfilled for arc (p, r_1) and (v_m, s_j), $1 \leq j \leq \ell$ since the considered vertex sets of G did not change. Thus, triple $(T', \mathcal{X}', \mathcal{W}')$ is a directed tree-decomposition of G.

The width of $(T', \mathcal{X}', \mathcal{W}')$ is at most the width of $(T, \mathcal{X}, \mathcal{W})$ since for every r_j, $1 \leq j \leq m$, then it holds that $|W'_{r_j} \cup \bigcup_{e \sim r_j} X'_e| \leq |W_r \cup \bigcup_{e \sim r} X_e|$.

If we perform this transformation for every $r \in V_T$ such that $|W_r| > 1$, we obtain a directed tree-decomposition of G which fulfills the properties of the lemma. □

Remark 3.3.10. By considering the directed tree-width forbidding empty sets W_r in [JRST01b] the statement of Lemma 3.3.9 can be strengthened to $|W_r| = 1$ for every $r \in V_T$.

Lemma 3.3.11. *Let G be a digraph, then the directed tree-width of G is the maximum directed tree-width of its strong components.*

Proof. The proof can be done similar to the proof of Lemma 3.2.8 using the result in [GR18] for directed tree-width $\text{d-tw}(G_1 \ominus G_2) = \max\{\text{d-tw}(G_1), \text{d-tw}(G_2)\}$, the statement of the lemma follows. □

For undirected tree-width, it is possible to define it by so called cops and robber games. In [JRST01b], the authors define a form of directed cops and robber game which is is strongly related to the definition of directed tree-width. We will consider these games in section 3.3.10.

Strong Component Directed Tree-Width

In the book "Quantitative Graph Theory" [DES14] in Chapter 6, Ordyniak and Kreutzer define a version of directed tree-width using strong components, which Ordyniak has first introduced in his thesis. They note that this definition that we call strong component directed tree-width differs from arboreal directed tree-width, the original one in [JRST01b], but only in a constant factor.

A *strong component* of a graph G is a maximum induced subdigraph of G which is strongly connected.

Definition 3.3.12 (strong component directed tree-width(sc-d-tw))**.** Let G be a digraph. A strong component directed tree-decomposition is a triple $(T, \mathcal{X}, \mathcal{W})$ such that $T = (V_T, E_T)$ is an out-tree, $\mathcal{X} = \{X_e \mid e \in E_T\}$ and $\mathcal{W} = \{W_r \mid r \in V_T\}$ are sets of subsets of V_G. Further the following conditions hold true:

(scdtw-1) $\mathcal{W} = \{W_r \mid r \in V_T\}$ is a partition of V_G

(scdtw-2) For all $(u, v) \in E_T$ the set $W_{\geq v} = \bigcup_{\tilde{v} \geq v} W_{\tilde{v}}$ is a strong component of $G - X_{(u,v)}$.

(scdtw-3) For all $v \in V_T$ it is $W_{\geq v} \cap \bigcup_{e\tilde{v}} X_e = \emptyset$

The *width* of a strong component directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ is

$$\max_{r \in V_T} |W_r \cup \bigcup_{e \sim r} X_e|.$$

The *strong component directed tree-width* of G, sc-d-tw(G) for short, is the smallest integer k such that there is a strong component directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of G of width k.

Please note, that in this definition the width is obtained very similar from the decomposition tree like for d-tw, but it lacks of adding "-1". This is only a constant factor, but for example though for a DAG D it holds that d-tw$(D) = 0$, it is sc-d-tw$(D) = 1$.

Example 3.3.13. The aboreal tree-decomposition given in Figure 3.2 is also a strong component directed tree-decomposition.

Strong Guards Directed Tree-Width

In Chapter 9 of [BJG18], Kreutzer and Kwon define a version of directed tree-width using strong guards.

For a digraph $G = (V, E)$ and two sets $X, Y \in V$ it holds that Y strongly guards X (or Y is a strong guard of X) if every directed walk starting and ending in X which contains a vertex of $V \setminus X$ also contains a vertex of Y. In other words, $X \setminus Y$ is the union of the vertex sets of some set of strong components of $G \setminus Y$.

Y weakly guards X (or Y is a weak guard of X) if every edge $e = (u, v) \in E$ with $u \in X \setminus Y$ has $v \in X \cup Y$.

Definition 3.3.14 (strong guards directed tree-width(sg-d-tw))**.** Let G be a digraph. A strong guard directed tree-decomposition is a triple $(T, \mathcal{X}, \mathcal{W})$ such that $T = (V_T, E_T)$ is an out-tree, $\mathcal{X} = \{X_e \mid e \in E_T\}$ and $\mathcal{W} = \{W_r \mid r \in V_T\}$ are sets of subsets of V_G Further the following conditions hold true:

(sgdtw-1) $\mathcal{W} = \{W_r \mid r \in V_T\}$ is a partition of V_G into non-empty sets

(sgdtw-2) For all $(u, v) \in E_T$ the set $X_{(u,v)}$ is a strong guard of $W_{\geq v} = \bigcup_{\tilde{v} \geq v} W_{\tilde{v}}$.

The *width* of a strong guard directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ is

$$\max_{r \in V_T} |W_r \cup \bigcup_{e \sim r} X_e| - 1.$$

The *strong guards directed tree-width* of G, sg-d-tw(G) for short, is the smallest integer k such that there is a strong component directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of G of width k.

Note that this definition is very similar to the one using Z-regularity and therefore it differs only in a constant factor to the aboreal directed tree-width definition.

Example 3.3.15. The aboreal tree-decomposition given in Figure 3.2 is also a strong guards directed tree-decomposition.

Reed's Directed Tree-Width

In [Ree99], Reed suggested an apparently very different definition of directed tree-width. But though the definition differs from the aboreal directed tree-decomposition by Johnson et al., the obtained widths differ by at most one.

The biggest difference is, that in contrary to all other directed tree-width definition, Reed does not create a decomposition tree with bags for every vertex, but only for the leaves of this tree.

Definition 3.3.16 (Reed's directed tree-width(r-d-tw))**.** Let G be a digraph. A Reed's directed tree-decomposition is a triple $(T, \mathcal{X}, \mathcal{W})$ such that $T = (V_T, E_T)$ is an out-tree, $\mathcal{X} = \{X_e \mid e \in E_T\}$ and $\mathcal{W} = \{W_r \mid r \text{ is a leaf of } V_T\}$ are sets of subsets of V_G Further the following conditions hold true:

(rdtw-1) $\mathcal{W} = \{W_r \mid r \text{ is a leaf of } V_T\}$ is a partition of V into subsets of size 1 (i.e. there is a bijection between the leaves of T and the vertices of G)

(rdtw-2) For all $(u, v) \in E_T$ the set $X_{(u,v)}$ is a cutset of $W_{\geq v} = \{W_{\tilde{v}} \mid \tilde{v} \geq v \text{ is a leaf of } V_T\}$ and $V \setminus W_{\geq v}$.

The *width* of a Reed's directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ is

$$\max_{r \in V_T} |\bigcup_{e \sim r} X_e| - 1.$$

The *Reed's directed tree-width* of G, r-d-tw(G) for short, is the smallest integer k such that there is a strong component directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of G of width k.

In his paper, Reed says that it is straightforward to show, that this definition is equal to the definition of aboreal directed tree-width by Johnson et al., if we modify the width such that it is obtained by $\max_{r \in V_T} |W_r \cup \bigcup_{e \sim r} X_e|$. However, without this modification, it just differs by ± 1.

Note that Reed did not define what he meant by *cutset.* However, by the equality to the definition of aboreal tree-width by Johnson et al., we can assume that he meant the same as they do by Z-regularity.

We can be sure, that not Z-normality is meant instead: Let $S_{1,n} = (V, E)$ be a star graph on $1 + n$ vertices, i.e. $V = \{v_0, v_1, \ldots, v_n\}$ and $E = \{\{v_0, v_i\} \mid 1 \leq i \leq n\}$. Further, let G_n be the complete biorientation of $S_{1,n}$, which is a directed co-graph. Then, tw$(S_{1,n}) = 1$ and further we know d-pw(G_n) = d-tw$(G_n) \leq 1$. But in any possible Reed's tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ for G_n there is a leaf u of T such that

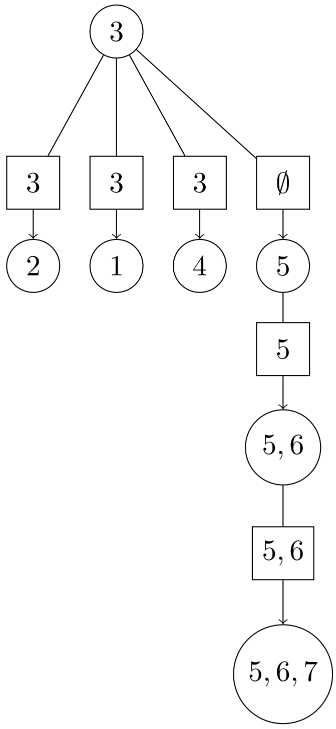


Figure 3.3: A Reed's directed tree-decomposition for graph G given in Figure 3.2.

$W_u = \{v_0\}$ and there is some $u' \in V_T$, such that $(u', u) \in E_T$. If $X_{(u',u)}$ would be $\{v_0\}$-normal, this would include that those two sets are disjoint. This would lead to $X_{(u',u)} = \{v_1, \ldots, v_n\}$ which implies a Reed's directed tree-width of $n - 1$. So in this case, Reed's directed tree-width and the aboreal directed tree-width would not be comparable. Thus, we can assume that regularity is meant, which would allow $X_{(u',u)} = \{v_0\}$ and therefore lead to a constant Reed's tree-width.

Note that in [Sei11], the author proves that the aboreal directed tree-width of [JRST01b] is not equal to the one defined by Reed. This only holds because the authors there used normality instead of regularity and therefore did not use the same definition as Reed himself in [Ree99].

At that point, please note that using normality or regularity does make a huge difference in Reed's directed tree-width, so that both variants are not comparable anymore. However, this does not make such a big difference for the aboreal directed tree-width, as this definition allows a rearrangement of the bags such that there is no bijection from the leafs of T to the vertices of G.

3.3.2 Directed feedback vertex set number

The directed feedback vertex set number (or DFVS-number for short) is probably the oldest of the measures considered here and was already considered by Karp in the 70s [Kar72].

Definition 3.3.17 (Directed feedback vertex set number)**.** The *directed feedback vertex set number* of a digraph $G = (V, E)$, denoted by $\text{fvs}(G)$, is the minimum cardinality of a set $S \subset V$ such that $G[V \setminus S]$ is a DAG.

Example 3.3.18.

1. A DAG has directed feedback vertex set number 0.
2. A bioriented path $\overleftrightarrow{P_n}$ has directed feedback vertex set number $\lfloor \frac{n}{2} \rfloor$
3. A cycle $\overrightarrow{C_n}$ has directed feedback vertex set number 1.
4. $\overleftrightarrow{K_n}$ has directed feedback vertex set number $n - 1$.

3.3.3 Directed feedback arc set number

Finding the directed feedback arc set number is a very fundamental problem and has applications in layered graph drawing.

Definition 3.3.19 (Directed feedback arc set number)**.** The *directed feedback arc set number* of a digraph $G = (V, E)$, denoted by $\text{fas}(G)$, is the minimum cardinality of a set $S \subset E$ such that $(V, E \setminus S)$ is a DAG.

Example 3.3.20.

1. A DAG has directed feedback arc set number 0.
2. A bioriented path $\overleftrightarrow{P_n}$ has directed feedback arc set number $n - 1$
3. A cycle $\overrightarrow{C_n}$ has directed feedback arc set number 1.
4. $\overleftrightarrow{K_n}$ has directed feedback arc set number $\frac{n(n-1)}{2}$.

3.3.4 Cycle Rank

Cycle rank was introduced in [Egg63] and also appeared in [Coh68] and [McN69].

Definition 3.3.21 (Cycle rank)**.** The *cycle rank* of a digraph $G = (V, E)$, denoted by $\text{cr}(G)$, is defined as follows.

- If G is acyclic, $\text{cr}(G) = 0$.
- If G is strongly connected, then $\text{cr}(G) = 1 + \min_{v \in V} \text{cr}(G - \{v\})$.
- Otherwise the cycle rank of G is the maximum cycle rank of any strongly connected component of G.

Results on the cycle rank can be found in [Gru08, Gru12]. In this papers Gruber proved the hardness of computing cycle rank, even for sparse digraphs of maximum outdegree at most 2.

Example 3.3.22.

1. A DAG has cycle rank 0.
2. A bioriented path $\overleftrightarrow{P_n}$ has cycle rank $\lfloor \log(n) \rfloor$
3. A cycle $\overrightarrow{C_n}$ has cycle rank 1.
4. $\overleftrightarrow{K_n}$ has cycle rank $n - 1$.

3.3.5 DAG-depth

The DAG-depth of a digraph was introduced in [GHK$^+$09] and motivated by tree-depth for undirected graphs, given in [NdM06].

For a digraph $G = (V, E)$ and $v \in V$, let G_v denote the subdigraph of G induced by the vertices which are reachable from v. The maximal elements in the partially ordered set $\{G_v \mid v \in V\}$ w.r.t. the digraph inclusion order (subdigraph) are the reachable fragments of G and will be denoted by $R(G)$.[4]

Definition 3.3.23 (DAG-depth)**.** Let $G = (V, E)$ be a digraph. The DAG-depth of G, denoted by $\mathrm{ddp}(G)$, is defined as follows.

- If $|V| = 1$, then $\mathrm{ddp}(G) = 1$.
- If G has a single reachable fragment, then $\mathrm{ddp}(G) = 1 + \min_{v \in V} \mathrm{ddp}(G - \{v\})$.
- Otherwise, $\mathrm{ddp}(G)$ equals the maximum over the DAG-depth of the reachable fragments of G.

We introduce a decomposition for DAG-depth, which is very similar to the one for cycle rank in [Gru12, McN69].

Definition 3.3.24 (Directed Elimination Forest)**.** A *directed elimination tree* for a digraph $G = (V, E)$ with $|R(G)| = 1$ reachable fragment is a rooted tree $T = (V_T, E_T)$ having the following properties.

1. $V_T \subseteq V \times 2^V$ and if $(x, X) \in V_T$, then $x \in X$.
2. The root of T is (v, V) for some $v \in V$.
3. If there is some vertex $(x, X) \in V_T$, then there is no vertex $(y, X) \in V_T$ for $x \neq y$.
4. If there is some vertex $(x, X) \in V_T$, and $G[X] - \{x\}$ has j reachable fragments $G_1 = (X_1, E_1), \ldots, G_j = (X_j, E_j)$, then (x, X) has exactly j children $(x_1, X_1), \ldots, (x_j, X_j)$ for $x_1, \ldots, x_j \in V$.

A *directed elimination forest* for some digraph G with $|R(G)| = j$ reachable fragments $G_1, \ldots, G_j$, is a rooted forest consisting of directed elimination trees for $G_1, \ldots, G_j$.

For some rooted tree T the *height* $h(T)$ is the number of edges on a longest path between the root and a leaf. For some forest F of rooted trees the height $h(F)$ is the maximum height of its trees.

Observation 3.3.25. For a digraph G the DAG-depth can be determined as follows:

$$\mathrm{ddp}(G) = 1 + min\{h(F) \mid \text{F is a directed elimination forest for G}\}.$$

Example 3.3.26.

1. A bioriented path $\overleftrightarrow{P_n}$ has dag-depth $\lfloor \log(n) \rfloor + 1$
2. $\overleftrightarrow{K_n}$ has DAG-depth n.

[4] In the undirected case, reachable fragments coincide with connected components.

3.3.6 DAG-width

The DAG-width has been defined in [BDHK06, BDH+12, Obd06]. While all variants of directed tree-width use a tree as a decomposition, the DAG-width uses a directed acyclic graph (DAG).

Let $G = (V_G, E_G)$ be an acyclic digraph. The partial order $\preccurlyeq_G$ on G is the reflexive, transitive closure of E_G. A *source* or *root* of a set $X \subseteq V_G$ is a $\preccurlyeq_G$-minimal element of X, that is, $r \in X$ is a root of X if there is no $y \in X$, such that $y \preccurlyeq_G r$ and $y \neq x$. Analogously, a *sink* or *leaf* of a set $X \subseteq V_G$ is a $\preccurlyeq_G$-maximal element.

Let $V' \subseteq V_G$, then a set $W \subseteq V_G$ *guards* V' if for all $(u, v) \in E_G$ it holds that if $u \in V'$ then $v \in V' \cup W$.

Definition 3.3.27 (DAG-width)**.** A *DAG-decomposition* of a digraph $G = (V_G, E_G)$ is a pair $(D, \mathcal{X})$ where $D = (V_D, E_D)$ is a directed acyclic graph (DAG) and $\mathcal{X} = \{X_u \mid X_u \subseteq V_G, u \in V_D\}$ is a family of subsets of V_G such that:

(dagw-1) $\bigcup_{u \in V_D} X_u = V_G$.

(dagw-2) For all vertices $u, v, w \in V_D$ with $u \succcurlyeq_D v \succcurlyeq_D w$, it holds that $X_u \cap X_w \subseteq X_v$.

(dagw-3) For all edges $(u, v) \in E_D$ it holds that $X_u \cap X_v$ guards $X_{\succcurlyeq v} \setminus X_u$, where $X_{\succcurlyeq v} = \cup_{v \succcurlyeq_D w} X_w$. For any source u, $X_{\succcurlyeq u}$ is guarded by $\emptyset$.

The *width* of a DAG-decomposition $(D, \mathcal{X})$ is the number

$$\max_{u \in V_D} |X_u|.$$

The *DAG-width* of a digraph G, $\text{dagw}(G)$ for short, is the smallest width of all possible DAG-decompositions for G.

Note that the definition of DAG-width does not contain a -1, which differs from the other presented directed width measures. We will notice this again in the comparison of several width parameters: There we will see, that this leads to a difference of at least $+1$ to many other width parameters, even for small and trivial examples.

Example 3.3.28. In Figure 3.4 we show an illustration of a DAG-decomposition for a digraph G, see Figure 3.2.

We use the restriction to nice DAG-decompositions from [BDH+12].

Definition 3.3.29 (Nice DAG-decomposition, [BDH+12])**.** A DAG-decomposition $(D, \mathcal{X})$ of a digraph G is *nice*, if the following properties are fulfilled.

1. D has exactly one root r.
2. Every vertex in D has at most two successors.
3. If vertex d has two successors d' and d'', then it holds that $X_d = X_{d'} = X_{d''}$.

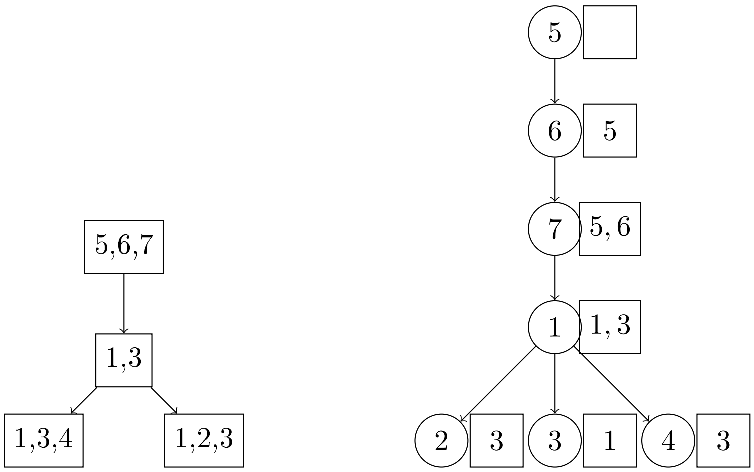


Figure 3.4: An illustration of a DAG-decomposition of DAG-width 3 (left) and a Kelly decomposition of width 3 (right) for digraph G, see Figure 3.2. In the Kelly decomposition, the round vertices represent the W sets, while the squares next to them represent the corresponding X sets.

4. If vertex d has one successors d', then it holds that $|X_d \triangle X_{d'}| = 1$. [5]

Lemma 3.3.30 ([BDH$^+$12]). *If digraph G has a DAG-decomposition of width k, it also has a nice DAG-decomposition of width k.*

There are even digraphs on n vertices whose optimal DAG-decompositions have super-polynomially many bags w.r.t n [AKR16]. Furthermore, it has been shown that deciding whether the DAG-width of a given digraph is at most a given value is PSPACE-complete [AKR16].

3.3.7 Kelly-width

Next, we consider the digraph width measure Kelly-width. Its definition is also based on the existence of a special DAG. While a DAG-decomposition has one vertex set for every vertex of the decomposition, within a Kelly-decomposition there are two vertex sets for every vertex of the decomposition. Kelly-width has been defined in [HK08].

Definition 3.3.31 (Kelly-width). A *Kelly decomposition* of a digraph $G = (V_G, E_G)$ is a triple $(\mathcal{W}, \mathcal{X}, D)$ where D is a directed acyclic graph, $\mathcal{X} = \{X_u \mid X_u \subseteq V_G, u \in V_D\}$ and $\mathcal{W} = \{W_u \mid W_u \subseteq V_G, u \in V_D\}$ are families of subsets of V_G such that:

(kw-1) $\mathcal{W}$ is a partition for V_G.

(kw-2) For all vertices $v \in V_G$, X_v guards $W_{\succcurlyeq v}$.

[5] We define $A \triangle B$ as the symmetric difference.

(kw-3) For all vertices $v \in V_G$, there is a linear order $u_1, \dots, u_s$ on the children of v such that for every u_i it holds that $X_{u_i} \subseteq W_i \cup X_i \cup \bigcup_{j<i} W_{\succcurlyeq u_j}$. Similarly, there is a linear order $r_1, r_2, \dots$ on the roots of D such that for each root r_i it holds that $W_{r_i} \subseteq \bigcup_{j<i} W_{\succcurlyeq r_j}$.

The *width* of a Kelly decomposition $(\mathcal{W}, \mathcal{X}, D)$ is the number

$$\max_{u \in V_D} |X_u| + |W_u|.$$

The *Kelly-width* of a digraph G, denoted with $\text{kw}(G)$, is the smallest width of all possible Kelly decompositions for G.

Example 3.3.32. In Figure 3.4 we show an illustration of a Kelly decomposition for a digraph G, see Figure 3.2.

We will use the following notation of a directed elimination ordering.

Definition 3.3.33 (Directed Elimination Ordering)**.** Let $G = (V, E)$ be a digraph. A directed elimination ordering $\lhd$ on G is a linear ordering on V. For $\lhd = (v_0, v_1, \dots, v_{n-1})$ we define

- $G_0^{\lhd} = G$
- $G_{i+1}^{\lhd} = (V_{i+1}^{\lhd}, E_{i+1}^{\lhd})$ with $V_{i+1}^{\lhd} = V_i^{\lhd} \setminus \{v_i\}$ and $E_{i+1}^{\lhd} = \{(u, v) \mid (u, v) \in E_i^{\lhd} \text{ and } u, v \neq v_i \text{ or } (u, v_i), (v_i, v) \in E_i^{\lhd}, u \neq v\}$

$G_i^{\lhd}$ is the directed elimination graph at step i according to $\lhd$.

The *width* of $\lhd$ is the maximum out-degree of v_i in $G_i^{\lhd}$ over all i.

Lemma 3.3.34 ([HK08])**.** *Let G be a digraph. The following are equivalent:*

1. *G has Kelly-width at most $k + 1$*
2. *G has a directed elimination ordering of width $\leq k$*

3.3.8 Directed NLC-width

The NLC-width (nlcw) for undirected graphs was introduced in [Wan94]. We now give a directed version of this parameter, similar to the directed linear NLC-width given in [GR19a].

Definition 3.3.35 (directed NLC-width)**.** The class of directed NLC-width at most k, $d-\text{NLC}_k$ for short, is recursively defined as follows

1. Creation of a new vertex with label a, denoted by $\bullet_a$ for some $a \in [k]$ is in $d - \text{NLC}_k$.

2. Disjoint union of two vertex-disjoint labeled digraphs $G = (V_G, E_G, \text{lab}_G), H = (V_H, E_H, \text{lab}_H) \in d - \text{NLC}_k$ with two relations $\overrightarrow{S}, \overleftarrow{S} \in [k]^2$, denoted by

$$G \otimes_{(\overrightarrow{S}, \overleftarrow{S})} H := (V', E', \text{lab}'),$$

   where $V' := V_G \cup V_H$,

$$\begin{aligned} E' := E_G \cup E_H \quad & \cup \; \{(u,v) \mid u \in V_G,\ v \in V_H,\ (\text{lab}_G(u), \text{lab}_H(v)) \in \overrightarrow{S}\} \\ & \cup \; \{(v,u) \mid u \in V_G,\ v \in V_H,\ (\text{lab}_G(u), \text{lab}_H(v)) \in \overleftarrow{S}\}, \end{aligned}$$

   and

$$\text{lab}'(u) \ := \ \begin{cases} \text{lab}_G(u) & \text{if } u \in V_G \\ \text{lab}_H(u) & \text{if } u \in V_H \end{cases}$$

   for every $u \in V'$ is in $d - \text{NLC}_k$.

3. Change every label a in $G = (V, E, \text{lab})$ into label $R(a)$ by some function $R : [k] \to [k]$ denoted by $\circ_R := (V, E, \text{lab}')$ with

$$\text{lab}'(u) := R(\text{lab}_G(u))$$

   for every $u \in V_G$ is in $d - \text{NLC}_k$.

The *directed NLC-width* of an unlabeled digraph $G = (V, E)$ is the smallest integer k, such that there is a mapping $\text{lab} : V \to [k]$ such that the labeled digraph (V, E, lab) is in $d - \text{NLC}_k$.

An expression X built with the operations defined above is called a *directed NLC-width k-expression.*

Example 3.3.36.

1. A bioriented path $\overleftrightarrow{P_n}$ has directed NLC-width at most 3.
2. A cycle $\overrightarrow{C_n}$ has directed NLC-width at most 4.
3. $\overleftrightarrow{K_n}$ has directed NLC-width 1.
4. An out-tree has directed NLC-width at most 3.

3.3.9 Directed Clique-Width

Directed clique-width has been introduced together with clique-width on undirected graphs by Courcelle in [CO00]. The linear clique-width for undirected graphs was introduced in [GW05b] as a parameter by restricting the clique-width, to an underlying path-structure.

Definition 3.3.37 (directed clique-width)**.** The class of directed clique-width at most k, $d - \text{CW}_k$ for short, is recursively defined as follows:

1. Creation of a new vertex with label a, denoted by $\bullet_a$, for some $a \in [k]$ is in $d - \text{CW}_k$.

2. Disjoint union of two vertex-disjoint labeled digraphs $G = (V_G, E_G, \text{lab}_G), H = (V_H, E_H, \text{lab}_H) \in d - \text{CW}_k$, denoted by $G \oplus H$ where

$$G \oplus H = (V', E', \text{lab}')$$

defined by $V' := V_G \cup V_H$, $E' := E_G \cup E_H$, and

$$\text{lab}'(u) \ := \begin{cases} \text{lab}_G(u) & \text{if } u \in V_G \\ \text{lab}_H(u) & \text{if } u \in V_H \end{cases}$$

for every $u \in V'$ is in $d - \text{CW}_k$.

3. For $G = (V, E, \text{lab}) \in d - \text{CW}_k$, inserting an arc from every vertex with label a to every vertex with label b, where $a, b \in [k]$, $a \neq b$, denoted by $\alpha_{a,b} := (V, E', \text{lab})$ with
$$E' := E \cup \{(u, v) \mid u, v \in V,\ u \neq v,\ \text{lab}(u) = a,\ \text{lab}(v) = b\}$$
is in $d - \text{CW}_k$.

4. For $G = (V, E, \text{lab}) \in d - \text{CW}_k$, change label a into label b, denoted by $\rho_{a \to b} = (V, E, \text{lab}')$ with
$$\text{lab}'(u) \ := \begin{cases} \text{lab}(u) & \text{if } \text{lab}(u) \neq a \\ b & \text{if } \text{lab}(u) = a \end{cases}$$
for every $u \in V_G$ is in $d - \text{CW}_k$.

The *directed clique-width* of an unlabeled digraph $G = (V, E)$, d-cw(G) for short, is the smallest integer k, such that there is a mapping lab : $V \to [k]$ such that the labeled digraph (V, E, lab) is in $d - \text{CW}_k$.

An expression X built with the operations defined above is called a *directed (linear) clique-width k-expression.*

Example 3.3.38. 1. A bioriented path $\overleftrightarrow{P_n}$ has directed clique-width at most 3.

2. A cycle $\overrightarrow{C_n}$ has directed clique-width at most 4.

3. $\overleftrightarrow{K_n}$ has directed clique-width 2.

4. An out-tree has directed clique-width at most 3.

3.3.10 Cops and Robbers Games

A cops and robbers game on a (directed) graph is a pursuit-evasion game with two teams of players, the cops and the robbers, moving from vertex to vertex along the arcs/edges of a graph. The cops try to "catch" the robbers by moving onto the vertices where the robbers are positioned, while the robbers try to evade this capture. But cops and robbers games with varying rules do not only model a range of pursuit-evasion games, but relate to useful graph parameters. In the undirected case, there are equivalent graph parameters for path-width and tree-width [Bod98, ST93]. We analyzed some special undirected cops and robbers game in [MRW20]. For directed graphs, there are also some known relations, for example for directed path-width [Bar06], directed tree-width [JRST01b], DAG-width [BDHK06] or Kelly-width [HK08]. For the survey in this chapter, we cite from Chapter 6 of [DES14].

Let $G = (V, E)$ be a directed graph with one robber and a set of cops. A position in the game is a pair (C, r) where $C \subseteq V$ is the current position of the cops and $r \in V$ is the current position of the robber. Initially, there is no cop on the graph, i.e. $C_0 = \emptyset$ and in the first round the robber can choose a start position r_0. In every round $i + 1$, (C_i, r_i) is the current position of the cops and robber. The game is then played as follows: The cops give their new position C_{i+1}. Then the robber can chose any vertex r_{i+1} as a new position, that is reachable from r_i in the graph $G - (C_i \cap C_{i+1})$. In the undirected case, reachability is very clear. In the directed case, there are two variations of reachability: In (strong) component searching, the robber can move to every vertex in the same strong component of $G - (C_i \cap C_{i+1})$. In reachability searching, the robber can move to any vertex r_{i+1} such that there is a directed walk from r_i to r_{i+1}. If $r_i \in C_i$ after any round i, then the cops capture the robber and win the game. Otherwise, the game never ends and the robber wins the game. Clearly, the game can allways be won by the cops, by positioning a cop on every vertex of G. However, an interesting question ist, how many cops are needed for a graph G, such that there is always a winning strategy for the cops.

By varying the rules, many different cops and robber games can be defined. The best known modification is, if the cops know the current robber position (visible CnR-Game) or do not know the current robber position (invisible CnR-Game). Another variant is a so-called inert robber: This robber is only allowed to move, if $r_i \in C_{i+1}$, so if the robber would be captured in the next round.

In the undirected case, the minimal number of cops required to capture a robber in the invisible CnR-Game is exactly the path-width, in the visible CnR-Game exactly the tree-width of a graph.

An interesting subject in this context is monotonicity. A strategy for the cops is called monotone, if the cops never move twice to the same vertex, i.e. if any cop has been placed on a vertex v and then move this cop elsewhere, they are not allowed to move back to v. To define robber monotonicity, we first need to define robber space. At any position (C_i, r_i) of the game let R_i be the set of nodes reachable by the robber from r_i, if he were allowed to move with all cops remaining on C_i. This means, $R_i = \{v \mid \text{ there is a directed path in } V - C_i \text{ from } r_i \text{ to } v\}$. A strategy is called robber monotone if in every play $(C_0, r_0), \ldots, (C_i, r_i), \ldots$ the robber space is nonincreasing, i.e. $R_{i+1} \subseteq R_i$ for all i. A game is called robber monotone, if whenever k cops suffice to catch a robber on a graph G, then k cops have a robber-monotone winning strategy. The game is called cop montone, if the analogous condition holds for cop-monotone winning strategies.

While in the undirected case many game variants turn out to be monotone (that is robber and cop monotone), especially the game variants equal to path-width and tree-width, this is quite different for directed gaphs. The invisible reachability cops and robber game on digraphs is monotone, but the strong component game with visible robber is not cop monotone and not robber monotone, but has bounded robber monotonicity: Whenever k cops have a winning strategy in the game, then $3k$ cops have a robber-monotone winning strategy. The reachability game with visible robber is neither cop nor robber monotone and further, it is unknown whether the monotonicity cost can be bounded.

In the following proposition we give the relation between several directed width parameters and directed cops and robber games. They are also taken from [DES14] and [BJG18].

Proposition 3.3.39. *Let $G = (V, E)$ be a directed graph.*

- *G has directed path-width k if and only if k cops have a (monotone) winning strategy for the invisible reachability cops and robber game on G.*
- *If G has directed tree-width k, then k cops have a robber monotone winning strategy in the visible strong component cops and robber game on G. If k cops have a winning strategy in this game, then the directed tree-width of G is at most $3k + 2$.*
- *G has DAG-width k if and only if k cops have a robber monotone winning strategy in the visible reachability cops and robber game on G.*
- *G has Kelly-width at most k if and only if k cops have a cop monotone winning strategy for an inert robber in the invisible reachability cops and robber game on G.*

3.4 Directed Coloring

Graph coloring is one of the basic problems in graph theory, which has already been considered in the 19th century. A k-coloring for an undirected graph G is a k-labeling of the vertices of G such that no two adjacent vertices have the same label. The smallest k such that a graph G has a k-coloring is named the chromatic number of G. Thus, by definition, the chromatic number is a graph parameter, as it assigns an integer to every graph. However, as this does not lead to a structure of the graph, like a tree or a path, one could not speak of a width measure for graphs. As we consider graph parameters in general, but mainly digraph width measures in this work, we will only give a short view on directed graph coloring.

3.4.1 Oriented Coloring

Oriented coloring has been introduced much later by Courcelle [Cou94]. One could easily apply the definition of graph coloring to directed graphs. But as this would not take the direction of the arcs into account, this would not be very interesting. For such a definition, the coloring of a directed graph would be the coloring of the underlying undirected graph.

Oriented coloring also considers the direction of the arcs. An oriented k-coloring of an oriented graph $G = (V, A)$ is a partition of the vertex set V into k independent sets, such that all the arcs linking two of these subsets have the same direction. In the oriented chromatic number problem there is given some oriented graph G and some integer k and one has to decide whether there is an oriented k-coloring for G. As even this problem is NP-hard, finding the chromatic number of an oriented graph is NP-hard, like it is for so many other graph parameters.

Please note that huge parts of this chapter are taken from [GKR19b].

We now give some formal definitions.

Definition 3.4.1 (Oriented Graph Coloring [Cou94])**.** An *oriented k-coloring* of an oriented graph $G = (V, A)$ is a mapping $c : V \to \{1, \ldots, k\}$, such that:

- $c(u) \neq c(v)$ for every $(u, v) \in A$
- $c(u) \neq c(y)$ for every two arcs $(u, v) \in A$ and $(x, y) \in A$ with $c(v) = c(x)$

The *oriented chromatic number* of G, denoted by $\chi_o(G)$, is the smallest k, such that G has an oriented k-coloring. The vertex sets $V_i = \{v \in V \mid c(v) = i\}$, $1 \leq i \leq k$, divide a partition of V into so called *color classes*.

For two oriented graphs $G_1 = (V_1, A_1)$ and $G_2 = (V_2, A_2)$ a *homomorphism* from G_1 to G_2, $G_1 \to G_2$ for short, is a mapping $h : V_1 \to V_2$, such that $(u, v) \in A_1$ implies that $(h(u), h(v)) \in A_2$. The oriented graphs G_1 and G_2 are *homomorphically equivalent*, if there is a homomorphism from G_1 to G_2 and one from G_2 to G_1. A homomorphism from G_1 to G_2 can be regarded as an oriented coloring of G_1 that uses the vertices of G_2 as colors classes. This leads to equivalent definitions for oriented coloring and oriented chromatic number. There is an oriented k-coloring of an oriented graph G_1 if and only if there is a homomorphism from G_1 to some oriented graph G_2 on k vertices. That is, the oriented chromatic number of G is the minimum number of vertices in an oriented graph G_2, such that there is a homomorphism from G_1 to G_2. Obviously, G_2 can be chosen as a tournament.

Observation 3.4.2. There is an oriented k-coloring of an oriented graph G_1 if and only if there is a homomorphism from G_1 to some tournament G_2 on k vertices. Further, the oriented chromatic number of G is the minimum number of vertices in a tournament G_2, such that there is a homomorphism from G_1 to G_2.

Lemma 3.4.3. *Let G be an oriented graph and H be a subdigraph of G, then $\chi_o(H) \leq \chi_o(G)$.*

Example 3.4.4. For oriented paths and oriented cycles we know: $\chi_o(\overrightarrow{P_2}) = 2$, $\chi_o(\overrightarrow{P_3}) = 3$, $\chi_o(\overrightarrow{C_4}) = 4$, $\chi_o(\overrightarrow{C_5}) = 5$.

An oriented graph $G = (V, A)$ is an *oriented clique* (*o-clique*) if $\chi_o(G) = |V|$. Thus all graphs given in Example 3.4.4 are oriented cliques.

Name: Oriented Chromatic Number (OCN)

Instance: An oriented graph $G = (V, A)$ and a positive integer $c \leq |V|$.
Question: Is there an oriented c-coloring for G?

If c is constant and not part of the input, the corresponding problem is denoted by OCN_c. Even for DAGs OCN_4 is NP-complete [CD06].

The definition of oriented coloring is also used for undirected graphs. For an undirected graph G the maximum value $\chi_o(G')$ of all possible orientations G' of G is considered. In this sense, every tree has oriented chromatic number at most 3. For several further graph classes there exist bounds on the oriented number. Among these are outerplanar graphs [Sop97], planar graphs [Mar13], and Halin graphs [DS14].

3.4.2 Acyclic coloring of directed graphs

This section is taken from our paper [GKR21a].

We consider the approach for coloring digraphs given in [NL82]. A set V' of vertices of a digraph G is called *acyclic* if $G[V']$ is acyclic.

Definition 3.4.5 (Acyclic graph coloring [NL82])**.** An *acyclic r-coloring* of a digraph $G = (V, E)$ is a mapping $c : V \to \{1, \ldots, r\}$, such that the color classes $c^{-1}(i)$ for $1 \leq i \leq r$ are acyclic. The *dichromatic number* of G, denoted by $\vec{\chi}(G)$, is the smallest r, such that G has an acyclic r-coloring.

There are several works on acyclic graph coloring [BFJ$^+$04, Moh03, NL82] including several recent works [LM17, MSW19, SW20]. The following observations support that the dichromatic number can be regarded as a natural counterpart of the well known chromatic number $\chi(G)$ for undirected graphs G.

Observation 3.4.6. For every complete bioriented digraph G it holds thats $\vec{\chi}(G) = \chi(und(G))$.

Observation 3.4.7. For every directed graph G it holds that $\vec{\chi}(G) \leq \chi(und(G))$.

Observation 3.4.8. Let G be a digraph and H be a subdigraph of G, then $\vec{\chi}(H) \leq \vec{\chi}(G)$.

Name: Dichromatic Number (DCN)
Instance: A digraph $G = (V, E)$ and a positive integer $r \leq |V|$.
Question: Is there an acyclic r-coloring for G?

If r is a constant and not part of the input, the corresponding problem is denoted by r-Dichromatic Number (DCN_r). Even DCN_2 is NP-complete [FHM03].

3.5 Comparing Directed Width Parameters

3.5.1 Directed linear width and undirected linear width

Next we compare the directed width of a digraph G and the undirected width of its underlying undirected graph $und(G)$. Several of the following results are taken from [GR19a].

Theorem 3.5.1. *Let G be a directed graph.*

(a) $d\text{-}pw(G) \leq pw(und(G))$

(b) $d\text{-}cutw(G) \leq cutw(und(G))$

(c) $nw(und(G)) \leq d\text{-}nw(G) \leq \Delta(und(G)) \cdot nw(und(G))$

(d) $lnlcw(und(G)) \leq d\text{-}lnlcw(G) \leq \Delta(und(G)) \cdot lnlcw(und(G)) + 1$

(e) $lcw(und(G)) \leq d\text{-}lcw(G) \leq \Delta(und(G)) \cdot lcw(und(G)) + 1$

(f) $lrw(und(G)) \leq d\text{-}lrw(G) \leq \Delta(und(G)) \cdot 2^{lrw(und(G))+1} - 1$

(g) *[JRST01b]* $d\text{-}tw(G) \leq tw(und(G))$

Proof. (a) See Lemma 3.2.3.

(b) Let $G = (V, E)$ be a digraph and $und(G)$ be the underlying undirected graph of cut-width k. Let φ be the corresponding ordering of the vertices, such that for every i, $1 \leq i \leq |V|$ there are at most k edges $\{u, v\}$ such that $u \in L(i, \varphi, und(G))$ and $v \in R(i, \varphi, und(G))$. Since every undirected edge $\{u, v\}$ in $und(G)$ comes from a directed edge (u, v), a directed edge (v, u), or both, and the directed cut-width only counts edges directed forward, the same layout shows that the directed cut-width of G is at most k.

(c) Let $G = (V, E)$ be a digraph of directed neighbourhood-width k and $\varphi \in \Phi(G)$ a linear layout, such that for every $i \in [|V|]$, we have $|N(L(i, \varphi, G), R(i, \varphi, G))| \leq k$. Since for every pair of vertices in G of the same directed neighbourhood the corresponding vertices in $und(G)$ have the same neighbourhood, it follows that for every $i \in [|V|]$, we have $|N(L(i, \varphi, und(G)), R(i, \varphi, und(G)))| \leq k$. Thus, the neighbourhood-width of $und(G)$ is at most k.

Let $G = (V, E)$ be a digraph and $und(G) = (V, E_u)$ be the underlying undirected graph of neighbourhood-width k. Then there is a layout $\varphi \in \Phi(und(G))$, such that for every $1 \leq i \leq |V|$ the vertices in $L(i, \varphi, und(G))$ can be divided into at most k subsets $L_1, \ldots, L_k$, such that the vertices of set L_j, $1 \leq j \leq k$, have the same neighbourhood with respect to the vertices in $R(i, \varphi, und(G))$. One of these sets L_j may consist of vertices that have no neighbours $v \in R(i, \varphi, und(G))$. Every of the remaining sets L_j has at most $\Delta(und(G))$ vertices u such that there is an edge $\{v, u\} \in E_u$ with $v \in R(i, \varphi, und(G))$. Let $1 \leq i \leq |V|$.

- If there is one set L_j which consists of vertices that have no neighbours $v \in R(i, \varphi, und(G))$, then there are at most $\Delta(und(G)) \cdot (k-1)$ vertices $u \in L(i, \varphi, und(G))$, such that there is an edge $\{v, u\} \in E_u$ with $v \in R(i, \varphi, und(G))$.
- Otherwise there are at most $\Delta(und(G)) \cdot k$ vertices $u \in L(i, \varphi, und(G))$, such that there is an edge $\{v, u\} \in E_u$ with $v \in R(i, \varphi, und(G))$.

Thus for every $1 \leq i \leq |V|$ the vertices in $L(i, \varphi, G)$ can be divided into $k' \leq \Delta(und(G)) \cdot k$ subsets $L'_1, \ldots, L'_{k'}$, such that the vertices of set L'_j, $1 \leq j \leq k'$, have the same directed neighbourhood with respect to the vertices in $R(i, \varphi, G)$. Thus the directed neighbourhood-width of G is at most $\Delta(und(G)) \cdot k$.

(d) Let G be a digraph of directed linear NLC-width k and X be a directed linear NLC-width k-expression for G. A linear NLC-width k-expression $c(X)$ for $und(G)$ can recursively be defined as follows.

- Let $X = \bullet_t$ for $t \in [k]$. Then $c(X) = \bullet_t$.
- Let $X = \circ_R(X')$ for $R : [k] \to [k]$. Then $c(X) = \circ_R(c(X'))$.
- Let $X = X' \otimes_{(\overrightarrow{S}, \overleftarrow{S})} \bullet_t$ for $\overrightarrow{S}, \overleftarrow{S} \subseteq [k]^2$ and $t \in [k]$. Then $c(X) = c(X') \times_{\overrightarrow{S} \cup \overleftarrow{S}} \bullet_t$.

The second bound follows by

$$\begin{aligned} \text{d-lnlcw}(G) &\leq \text{d-nw}(G) + 1 \leq \Delta(und(G)) \cdot \text{nw}(und(G)) + 1 \\ &\leq \Delta(und(G)) \cdot \text{lnlcw}(und(G)) + 1, \end{aligned}$$

whereas the inequalities hold, respectively, by Lemma 3.5.8, (c), and [Gur06b].

(e) Let G be a digraph of directed linear clique-width k and X be a directed linear clique-width k-expression for G. A linear clique-width k-expression $c(X)$ for $und(G)$ can recursively be defined as follows.

- Let $X = \bullet_t$ for $t \in [k]$. Then $c(X) = \bullet_t$.
- Let $X = X' \oplus \bullet_t$ for $t \in [k]$. Then $c(X) = c(X') \oplus \bullet_t$.
- Let $X = \rho_{i \to j}(X')$ for $i, j \in [k]$. Then $c(X) = \rho_{i \to j}(c(X'))$.
- Let $X = \alpha_{i,j}(X')$ for $i, j \in [k]$. Then $c(X) = \eta_{i,j}(c(X'))$.

The second bound follows by

$$\begin{aligned} \text{d-lcw}(G) &\leq \text{d-nw}(G) + 1 \leq \Delta(und(G)) \cdot \text{nw}(und(G)) + 1 \\ &\leq \Delta(und(G)) \cdot \text{lcw}(und(G)) + 1. \end{aligned}$$

whereas the inequalities hold, respectively, by Lemma 3.5.8, (c), and [Gur06b].

(f) Let G be a digraph of directed linear rank-width k and (T, f) be a directed linear rank-decomposition for G of width k. Then (T, f) is also a linear rank-decomposition for $und(G)$. Let e be an edge of T. Let $N_{V_1}^{V_2} = (n_{ij})$ be the adjacent matrix defined over the two-element field GF(2) for partition $V_1 \cup V_2$. If for G two rows in $M_{A_e}^{B_e}$ are linearly dependent then for $und(G)$ these two rows in $N_{A_e}^{B_e}$ are also linearly dependent. Thus we conclude that $\text{rg}^{(2)}(N_{A_e}^{B_e}) \leq \text{rg}^{(4)}(M_{A_e}^{B_e})$ and thus linear rank-width of $und(G) \leq k$.

The second bound follows by

$$\begin{aligned}\text{d-lrw}(G) &\leq \text{d-nw}(G) \leq \Delta(und(G)) \cdot \text{nw}(und(G)) \\ &\leq \Delta(und(G)) \cdot 2^{\text{lrw}(und(G))+1} - 1,\end{aligned}$$

whereas the inequalities hold, respectively, by Lemma 3.5.9, (c), and [OS06, Proposition 6.3].

See also Lemma 3.3.6. This completes the proof. □

Remark 3.5.2. In Theorem 3.5.1(a) and (b) the directed path-width of some digraph cannot be used to give an upper bound on the path-width of $und(G)$. Any transitive tournament has directed path-width 0 but its underlying undirected graph has a path-width which corresponds to the number of vertices. Also by restricting the vertex degree this is not possible by an acyclic orientation of a grid. The same examples also show that the directed cut-width of some digraph cannot be used to give an upper bound on the cut-width of $und(G)$.

Remark 3.5.3. Theorem 3.5.1(a) and (b) show that for path-width and cut-width the values do not grow when going to the directed variant. This changes for the other four parameters, since the set of all tournaments has unbounded directed width while the corresponding undirected width of set of all complete graphs is bounded by a small constant.

The relations shown in Theorem 3.5.1 allow to imply the following values for the directed linear clique-width and directed neighbourhood-width of a k-power graph of a path.

Corollary 3.5.4. *(a) For $n \geq k(k+1)+2$, we have d-lcw$((\overrightarrow{P_n})^k) = k+2$.*

(b) For $n \geq k(k+1)+2$, we have d-nw$((\overrightarrow{P_n})^k) = k+1$.

Proof. For $n \geq k(k+1)+2$ we know from [HMP09] that the (undirected) linear clique-width of a k-power graph of a path on n vertices is exactly $k+2$.

(a) For $n \geq k(k+1)+2$ the first statement follows by

$$k+2 = \text{lcw}(und((\overrightarrow{P_n})^k)) \leq \text{d-lcw}((\overrightarrow{P_n})^k) \leq k+2,$$

whereas these equality and inequalities hold, respectively, by [HMP09], Theorem 3.5.1, and Example 3.2.16.

(b) For $n \geq k(k+1)+2$ the second statement follows by

$$\begin{aligned}k+1 &= \text{lcw}(und((\overrightarrow{P_n})^k)) - 1 \leq \text{d-lcw}((\overrightarrow{P_n})^k) - 1 \\ &\leq \text{d-nw}((\overrightarrow{P_n})^k) \leq k+1,\end{aligned}$$

whereas these equality and inequalities hold, respectively, by [HMP09], Theorem 3.5.1, Lemma 3.5.8, and Example 3.2.18.

This completes the proof. □

Comparing the undirected width of a graph G and the directed width of its complete biorientation $\overleftrightarrow{G}$ the following results hold.

Theorem 3.5.5. *For each width measure $\beta \in \{pw, cutw, nw, lnlcw, lcw, lrw\}$ and every undirected graph G, we have $\beta(G) = d\text{-}\beta(\overleftrightarrow{G})$.*

Proof.

- By Lemma 3.2.4, this holds for directed path-width.
- By Theorem 3.5.1(b) it remains to show that the cut-width of G is at most the directed cut-width of $\overleftrightarrow{G}$. Let $G = (V, E)$ be a graph and $\overleftrightarrow{G}$ its complete biorientation of directed cut-width k. Let φ be the corresponding ordering of the vertices, such that for every i, $1 \leq i \leq |V|$ there are at most k arcs (u, v) such that $u \in L(i, \varphi, und(G))$ and $v \in R(i, \varphi, und(G))$. Since every such arc corresponds to one undirected edge $\{u, v\}$ in G, the same layout shows that the cut-width of G is at most k.
- By Theorem 3.5.1(c) it remains to show that the directed neighbourhood-width of $\overleftrightarrow{G}$ is at most the neighbourhood-width of G. Let $\varphi \in \Phi(G)$ a linear layout, such that for every $i \in [|V|]$, we have $|N(L(i, \varphi, G), R(i, \varphi, G))| \leq k$. By the definitions of $\overleftrightarrow{G}$ and for neighbourhoods of directed graphs, it follows that for every $i \in [|V|]$ for the number of directed neighbourhoods, we have $|N(L(i, \varphi, \overleftrightarrow{G}), R(i, \varphi, \overleftrightarrow{G}))| \leq k$.
- By Theorem 3.5.1(d) it remains to show that the directed linear NLC-width of $\overleftrightarrow{G}$ is at most the linear NLC-width of G. Let X be an NLC-width k-expression for G. A directed NLC-width k-expression $c(X)$ for $\overleftrightarrow{G}$ can recursively be defined as follows.

  * Let $X = \bullet_t$ for $t \in [k]$. Then $c(X) = \bullet_t$.
  * Let $X = \circ_R(X')$ for $R : [k] \to [k]$. Then $c(X) = \circ_R(c(X'))$.
  * Let $X = X' \times_S X''$ for $S \subseteq [k]^2$. Then $c(X) = c(X') \otimes_{(S,S)} c(X'')$.

- By Theorem 3.5.1(e) it remains to show that the directed linear clique-width of $\overleftrightarrow{G}$ is at most the linear clique-width of G. Let X be a clique-width k-expression for G. A directed clique-width k-expression $c(X)$ for $\overleftrightarrow{G}$ can recursively be defined as follows.

  * Let $X = \bullet_t$ for $t \in [k]$. Then $c(X) = \bullet_t$.
  * Let $X = X' \oplus X''$. Then $c(X) = c(X') \oplus c(X'')$.
  * Let $X = \rho_{i \to j}(X')$ for $i, j \in [k]$. Then $c(X) = \rho_{i \to j}(c(X'))$.
  * Let $X = \eta_{i,j}(X')$ for $i, j \in [k]$. Then $c(X) = \alpha_{j,i}(\alpha_{i,j}(c(X')))$.

- By Theorem 3.5.1(f) it remains to show that the directed linear rank-width of $\overleftrightarrow{G}$ is at most the linear rank-width of G. Let (T, f) be a linear rank-decomposition of width k for G. Then (T, f) is also a linear rank-decomposition for $\overleftrightarrow{G}$. Let $N_{V_1}^{V_2} = (n_{ij})$ be the adjacent matrix defined over the two-element field GF(2) for partition $V_1 \cup V_2$. Since for every bioriented graph $N_{V_1}^{V_2} = M_{V_1}^{V_2}$ we conclude that the directed linear rank-width of $\overleftrightarrow{G}$ is at most k.

This completes the proof. □

It is already known that recognizing path-width ([ACP87]), cut-width ([Gav77]), linear NLC-width ([Gur06b]), linear clique-width ([FRRS09]), neighbourhood-width ([Gur06b]), and linear rank-width (by [Oum17] due [Kas08] and [Oum05]) are NP-hard. The results of Theorem 3.5.5 imply the same for the directed versions.

Corollary 3.5.6. *Given a digraph G and an integer k, then for every width measure $\beta \in \{$d-pw, d-cutw, d-nw, d-lnlcw, d-lcw, d-lrw$\}$, the problem to decide whether $\beta(G) \leq k$ is NP-complete.*

3.5.2 Linear width parameters

Relations between linear NLC-width, linear clique-width, neighbourhood-width, and linear rank-width

First we state the relation between the directed linear NLC-width and directed linear clique-width. The proofs can be done in the same way as for the undirected versions in [GW05b].

Lemma 3.5.7. *For every digraph G, we have*

$$\textit{d-lnlcw}(G) \leq \textit{d-lcw}(G) \leq \textit{d-lnlcw}(G) + 1.$$

Further there is also a very tight connection between the directed neighbourhood-width, directed linear NLC-width, and directed linear clique-width. The proofs of the following bounds can be done in a similar fashion as for the undirected versions in [Gur06b].

Lemma 3.5.8. *For every digraph G, we have*

$$\textit{d-nw}(G) \leq \textit{d-lnlcw}(G) \leq \textit{d-nw}(G) + 1$$

and

$$\textit{d-nw}(G) \leq \textit{d-lcw}(G) \leq \textit{d-nw}(G) + 1.$$

By the examples given in section 3.2 and simple observations, we conclude that every path $\overrightarrow{P_n}$, $n \geq 3$, has directed linear clique-width 3, paths $\overrightarrow{P_3}$ and $\overrightarrow{P_4}$ have directed linear NLC-width 2, every path $\overrightarrow{P_n}$, $n \geq 5$, has directed linear NLC-width 3, and every path $\overrightarrow{P_n}$, $n \geq 3$, has directed neighbourhood-width 2, which implies that the bounds of Lemma 3.5.7 and Lemma 3.5.8 cannot be improved.

Lemma 3.5.9. *For every digraph G, we have*

$$\textit{d-lrw}(G) \leq \textit{d-nw}(G).$$

Proof. Let G be a digraph with n vertices of directed neighbourhood-width k and $\varphi : V \to [n]$ be a layout such that $\text{d-nw}(\varphi, G) \leq k$. Using φ we define a caterpillar T_φ with consecutive pendant vertices $\varphi^{-1}(1), \ldots, \varphi^{-1}(n)$. Pair (T_φ, φ) leads to a directed linear rank decomposition for G. We want to determine the width of (T_φ, φ). Since for every i the vertices in $L(i, \varphi, G)$ define at most k neighbourhoods with respect to set $R(i, \varphi, G)$, every edge of T_φ leads to a partition of V into $L(i, \varphi, G)$ and $R(i, \varphi, G)$ for some i such that $M_{L(i,\varphi,G)}^{R(i,\varphi,G)}$ has at most k different rows and thus $\text{rg}(M_{L(i,\varphi,G)}^{R(i,\varphi,G)}) \leq k$. □

The following bound can be proved similarly to the case of clique-width and rank-width in [OS06, Proposition 6.3].

Lemma 3.5.10. *For every digraph G, we have*

$$\textit{d-lcw}(G) \leq 4^{\textit{d-lrw}(G)+1} - 1.$$

The shown bounds imply the following theorem.

Theorem 3.5.11. *Any two parameters in {d-nw, d-lnlcw, d-lcw, d-lrw} are equivalent.*

Theorem 3.5.12. *Any two parameters in {d-nw, d-lnlcw, d-lcw} are linearly equivalent.*

Using the arguments of [FOT10, Section 8] we obtain the next result.

Lemma 3.5.13. *There is some polynomial p such that for every digraph G, we have* $\textit{d-lcw}(G) \leq p(\Delta(G), \textit{d-lrw}(G))$.

Theorem 3.5.14. *For every class of digraphs $\mathcal{G}$ such that for all $G \in \mathcal{G}$ the value $\Delta(G)$ is bounded, any two parameters in {d-nw, d-lnlcw, d-lcw, d-lrw} are polynomially equivalent.*

Relations between cut-width and path-width

The directed path-width is even equal to the directed vertex separation number.

Lemma 3.5.15 ([YC08])**.** *For every digraph G, we have*

$$\textit{d-pw}(G) = \textit{d-vsn}(G).$$

In [FP13a] it is shown how to construct a directed path-decomposition of width twice the directed cut-width of the graph. Using the directed vertex separation number, we next show a better bound.

Lemma 3.5.16. *For every digraph G, we have*

$$d\text{-}pw(G) \leq d\text{-}cutw(G).$$

Proof. Let $G = (V, E)$ be a digraph of directed cut-width k. By (3.6) there is a layout $\varphi \in \Phi(G)$, such that for every $1 \leq i \leq |V|$ there are at most k arcs $(v, u) \in E$ such that $v \in R(i, \varphi, G)$ and $u \in L(i, \varphi, G)$. Thus for every $1 \leq i \leq |V|$ there are at most k vertices $u \in L(i, \varphi, G)$, such that there is an arc $(v, u) \in E$ with $v \in R(i, \varphi, G)$. Thus by (3.1) the directed vertex separation number of G is at most k and by Lemma 3.5.15 the directed path-width of G is at most k. □

The directed path-width and directed cut-width of a digraph can differ very much, e.g. a $\overleftrightarrow{K_{1,n}}$ has directed path-width 1 and directed cut-width $\lceil \frac{n}{2} \rceil$.

Lemma 3.5.17. *For every digraph G, we have*

$$d\text{-}cutw(G) \leq \min(\Delta^-(G), \Delta^+(G)) \cdot d\text{-}pw(G).$$

Proof. Let $G = (V, E)$ be a digraph of directed path-width k. By Lemma 3.5.15 and (3.1) there is a layout $\varphi \in \Phi(G)$, such that for every $1 \leq i \leq |V|$ there are at most k vertices $u \in L(i, \varphi, G)$, such that there is an arc $(v, u) \in E$ with $v \in R(i, \varphi, G)$. Thus for every $1 \leq i \leq |V|$ there are at most $\Delta^-(G) \cdot k$ arcs $(v, u) \in E$ such that $v \in R(i, \varphi, G)$ and $u \in L(i, \varphi, G)$. By (3.6) this implies that the directed cut-width of digraph G is at most $\Delta^-(G) \cdot k$.

The bound using Δ^+ instead of Δ^- can be shown in the same way using definition (3.3) instead of (3.1) and using definition (3.5) instead of (3.6). □

Theorem 3.5.18. *For every class of digraphs $\mathcal{G}$ such that for all $G \in \mathcal{G}$ the value $\min(\Delta^-(G), \Delta^+(G))$ is bounded any two parameters in $\{d\text{-}cutw, d\text{-}pw\}$ are linearly equivalent.*

Relations between path-width and neighbourhood-width

The directed neighbourhood-width and directed path-width of a digraph can differ very much, e.g. a $\overleftrightarrow{K_n}$ has directed neighbourhood-width 1 and directed path-width $n - 1$.

Lemma 3.5.19. *For every digraph G, we have*

$$d\text{-}pw(G) \leq \min(\Delta^-(G), \Delta^+(G)) \cdot d\text{-}nw(G).$$

Proof. Let $G = (V, E)$ be a digraph of directed neighbourhood-width k. Then there is a layout $\varphi \in \Phi(G)$, such that for every $1 \leq i \leq |V|$ the vertices in $L(i, \varphi, G)$ can be divided into at most k subsets $L_1, \ldots, L_k$, such that the vertices of set L_j, $1 \leq j \leq k$, have the same neighbourhood with respect to the vertices in $R(i, \varphi, G)$. Every of these sets L_j has at most $\Delta^-(G)$ vertices u such that there is an arc $(v, u) \in E$ with $v \in R(i, \varphi, G)$. Thus for every $1 \leq i \leq |V|$ there are at most $\Delta^-(G) \cdot k$ vertices

$u \in L(i, \varphi, G)$, such that there is an arc $(v, u) \in E$ with $v \in R(i, \varphi, G)$. Thus by (3.1) the directed vertex separation number of G is at most $\Delta^-(G) \cdot k$ and by Lemma 3.5.15 the directed path-width of G is at most $\Delta^-(G) \cdot k$.

The bound using Δ^+ instead of Δ^- can be shown in the same way using definition (3.3) instead of definition (3.1). □

The example $\overleftrightarrow{K_n}$ shows that the bound given in Lemma 3.5.19 is tight.

Lemmas 3.5.19, 3.5.8, and 3.5.10 imply the following bounds.

Corollary 3.5.20. *For every digraph G, we have*

$$\textit{d-pw}(G) \leq \min(\Delta^-(G), \Delta^+(G)) \cdot \textit{d-lnlcw}(G),$$

$$\textit{d-pw}(G) \leq \min(\Delta^-(G), \Delta^+(G)) \cdot \textit{d-lcw}(G), \textit{ and}$$

$$\textit{d-pw}(G) \leq \min(\Delta^-(G), \Delta^+(G)) \cdot (4^{\textit{d-lrw}(G)+1} - 1).$$

After considering the maximum vertex degree, we next make a stronger restriction by excluding all possible orientations of a $K_{\ell,\ell}$ as subdigraphs.

Corollary 3.5.21. *Let G be a digraph where $und(G)$ has no $K_{\ell,\ell}$ subgraph, then we have*

$$\textit{d-pw}(G) \leq \textit{pw}(\textit{und}(G)) \leq 2 \cdot \textit{lnlcw}(\textit{und}(G))(\ell - 1) \leq 2 \cdot \textit{d-lnlcw}(G)(\ell - 1).$$

Proof. By the results for undirected graphs in [Gur06b] we know that for every graph G which has no $K_{\ell,\ell}$ subgraph, we have

$$\text{pw}(G) \leq 2 \cdot \text{lnlcw}(G)(\ell - 1).$$

This implies for every digraph G, where $und(G)$ has no $K_{\ell,\ell}$ subgraph, we have

$$\text{pw}(\textit{und}(G)) \leq 2 \cdot \text{lnlcw}(\textit{und}(G))(\ell - 1).$$

Furthermore by Theorem 3.5.1(a) and Theorem 3.5.1(d) for every digraph G, where $und(G)$ has no $K_{\ell,\ell}$ subgraph, we have

$$\text{d-pw}(G) \leq \text{pw}(\textit{und}(G)) \leq 2 \cdot \text{lnlcw}(\textit{und}(G))(\ell - 1) \leq 2 \cdot \text{d-lnlcw}(G)(\ell - 1).$$

This completes the proof. □

Next we want to bound the directed linear clique-width in terms of the directed path-width.

Remark 3.5.22. For general digraphs and even for digraphs of bounded vertex degree the directed linear clique-width, directed linear NLC-width, directed neighbourhood-width, and directed linear rank-width cannot be bounded by the directed path-width by the following examples.

1. Let T' be an orientation of a tree, e.g. an out-tree or an in-tree. Then d-pw$(T') = 0$, as T' is a DAG. But d-lcw(T') is unbounded, since lcw$(und(T'))$ is unbounded [GW05b] and since lcw$(und(T')) \leq$ d-lcw(T') by Theorem 3.5.1.

2. Let G' be an acyclic orientation of a grid. Then d-pw$(G') = 0$, as G' is a DAG. But d-lcw(G') is unbounded, since lcw$(und(G'))$ is unbounded [GR00] and since lcw$(und(G')) \leq$ d-lcw(G') by Theorem 3.5.1.

3. The set of all k-power graphs of directed paths has directed path-width 0 (cf. Example 3.2.10) and directed linear clique-width $k + 2$ (Corollary 3.5.4).

Equivalent parameters

In Table 3.2 we summarize our results on the equivalence of linear width parameters for directed graphs. For general digraphs we have three classes of pairwise equivalent parameters, which reduces to two or one class for $\Delta(G)$ bounded or semicomplete $\Delta(G)$ bounded digraphs, respectively.

digraphs	equivalence	d-cutw	d-pw	d-lcw	d-lnlcw	d-nw	d-lrw
general	equivalent	•	•	•	•	•	•
	polynomially equivalent	•	•	•	•	•	
	linearly equivalent	•	•	•	•	•	
$\Delta(G)$ bounded	equivalent	•	•	•	•	•	•
	polynomially equivalent	•	•	•	•	•	•
	linearly equivalent	•	•	•	•	•	
semicomplete $\Delta(G)$ bounded	equivalent	•	•	•	•	•	•
	polynomially equivalent	•	•	•	•	•	•
	linearly equivalent	•	•	•	•	•	

Table 3.2: Classification of linear width parameters for directed graphs. The gray shades of the points represent sets of pairwise (linearly, polynomially) equivalent parameters.

3.5.3 Non-linear width parameters

There are also some known relations between non-linear width parameters. First of all, there is an equivalence of most definitions of directed tree-width. By [JRST01b, JRST01a, DES14, BJG18, Ree99] it follow that:

Theorem 3.5.23. *The following graph parameters are linearly equivalent:*

- *Aboreal directed tree-width d-tw*
- *Aboreal directed tree-width using Z-regularity instead of normality*
- *Strong-component directed tree-width sc-d-tw*
- *Strong guards directed tree-width sg-d-tw*
- *Reed's directed tree-width r-d-tw*

As all those parameters are equivalent, we will only consider d-tw in the following work. For this parameter we further get, by Theorem 3.5.1, g:

Theorem 3.5.24. *Let G be a digraph. Then d-tw$(G) \leq$ u-d-tw(G).*

In Table 3.3, we summarize some examples for the values of digraph width measures, which are considered in this work, for special digraphs. Further examples can be found in [GHK+14, Table 1].

G	d-tw(G)	d-pw(G)	fvs(G)	fas(G)	cr(G)	ddp(G)	dagw(G)	kw(G)	d-cw(G)
$\overrightarrow{P_n}$	0	0	0	0	0	$\lfloor\log(n)\rfloor+1$	1	1	3
$\overrightarrow{C_n}$	1	1	1	1	1	$\lfloor\log(n-1)\rfloor+2$	2	2	4
$\overrightarrow{T_n}$	0	0	0	0	0	n	1	1	3
$\overleftrightarrow{P_n}$	1	1	$\lfloor\frac{n}{2}\rfloor$	$n-1$	$\lfloor\log(n)\rfloor$	$\lfloor\log(n)\rfloor+1$	2	2	3
$\overleftrightarrow{K_n}$	$n-1$	$n-1$	$n-1$	$\frac{n(n-1)}{2}$	$n-1$	n	n	n	2

Table 3.3: The value of digraph width measures of special digraphs.

We now give some relationships between other non-linear (and linear) width parameters. The following proposition is taken of [GKRW21].

Proposition 3.5.25. *Let G be a digraph and $f, g \in \{$d-pw, d-tw, dagw, kw, cr, fvs, fas, d-lcw, d-cw$\}$. If $f(G) \leq k$, then $g(G) \leq h'_{f,g}(k)$ where $h'_{f,g}\colon \mathbb{N} \to \mathbb{N}$ is given by Table 3.4.*

Proof.

1. d-pw is unbounded in terms of kw: In [BJG18] the example of a complete biorientation of an undirected binary tree of height h is considered. This digraph has directed path-width h while it has the fixed Kelly-width of 2.

2. d-pw is unbounded in terms of d-tw: Holds with the example from 1 which is inspired by the undirected comparisons of path-width and tree-width. Increasing h, the directed tree-width is 1, while the directed path-width increases.

3. d-pw, d-tw, dagw and kw are unbounded in terms of d-lcw and thus in d-cw: The set of all bioriented cliques is a counterexample.

4. By [Gru12], cycle rank is an upper bound for directed path-width and thus for directed tree-width and further, cr plus 1 is an upper bound for dagw and kw.

5. The cycle rank can be much larger than directed path-width, directed tree-width, Kelly-width and DAG-width, which can be shown by a complete biorientation of a path graph $\overleftrightarrow{P_n}$ which has arbitrary large cycle rank $\lfloor\log(n)\rfloor$, see [McN69].

6. By [GHK+14], fvs is an upper bound of cr and thus the bounds from 4 are extendable to fvs.

g \ f	d-pw	d-tw	dagw	kw	cr	fvs	fas	d-lcw	d-cw
d-pw	k	∞	∞ [BDH+12]	∞	k	k	k	∞	∞
d-tw	k	k	$3k+1$ [BDH+12]	$6k-2$ [HK08]	k	k	k	∞	∞
dagw	$k+1$ [BDH+12]	∞ [BDH+12]	k	$72k^2$ [AKK+15]	$k+1$	$k+1$	$k+1$	∞	∞
kw	$k+1$ [GHK+14]	∞	??? [HK08]	k	$k+1$	$k+1$	$k+1$	∞	∞
cr	∞	∞	∞	∞	k	k	k	∞	∞
fvs	∞	∞	∞	∞	∞	k	k	∞	∞
fas	∞	∞	∞	∞	∞	∞	k	∞	∞
d-lcw	∞	∞	∞	∞	∞	∞	∞	k	∞
d-cw	∞	∞	∞	∞	∞	∞	∞	k	k

Table 3.4: Relations between digraph parameter on **digraphs**. The parameter of the left column is bounded by the respective parameter of the top row by the specified function where k is the corresponding width. We use '∞' if the relation is unbounded, that is if $h'_{f,g}$ does not exist. The cell with '???' represents the remaining relation of the conjecture on DAG-width and Kelly-width.

7. As fvs of a bioriented path is $\lfloor \frac{n}{2} \rfloor$, by the same reason as in 5, fvs can be much larger than d-tw, d-pw, kw and dagw. Further it can be much larger than cycle rank, which can be shown by the disjoint union of $\frac{n}{3}$ directed cycles $\overrightarrow{C_3}$ which has cycle rank 1 but arbitrary large DFVS-number $\frac{n}{3}$.

8. By definition, fas is an upper bound for fvs. Therefore, the results of 4 can be extended to fas.

9. As fas of a bioriented path is $n-1$, it can be much larger than d-pw, d-tw, dagw, kw as well as directed (linear) clique-width. Further, by the same Argument as in 7 it can be much larger than cycle rank.

10. fas can not be upper bounded by fvs. A counterexample can be given by a bioriented star: The directed feedback vertex set number of a bioriented star is 1, whereas the directed feedback arc set number is $n-1$.

11. d-cw and thus also d-lcw is unbounded in terms of d-pw, d-tw, dagw, kw, cr, fvs and fas. An acyclic orientation of a grid graph is a counterexample.

12. d-lcw is unbounded in terms of d-cw: Same example as in 1.

13. d-cw is bounded by d-lcw: This follows immediately from the definition.

□

Note that to compare DAG-width and Kelly-width, right now, only one direction is known. Till now it is only possible to bound Kelly-width by DAG-width. However, it is assumed that both directions are possible with a polynomial factor. We will prove later that equivalence holds on special directed graph classes.

Further note that directed DAG-depth is not included in Table 3.4. This is as most relations between ddp and other parameters are not known yet. However, by the fact that ddp is an upper bound for cr, the results of 4 in the previous proof are extandable to directed DAG-depth.

Furthermore, Proposition 3.5.25 contains in particular the known fact that directed path-width poses as an upper bound for all tree-width inspired width parameters. Moreover, on semicomplete digraphs, it also is an upper bound on directed clique-width. It therefore suffices, towards a proof of Theorem 5.6.6, to establish upper bounds on directed path-width in terms of directed tree-width, DAG-width, and Kelly-width, as well this can be extended to also include directed linear clique-width.

In Table 3.4 we give exact bounds for several directed non-linear graph parameters and the linear directed path-width. However, the exact values are not always necessary, sometime it suffices to know if the parameters are comparable at all. In Table 3.5 we summarize the known relations between some measures considered in this work.

$$\begin{array}{ccccccccccccc} & & & & & & & & \text{ddp} & & & & \\ & & & & & & & & \downarrow & & & & \\ \text{d-tw} & \xleftarrow{[\text{BDH}^+12]} & \text{dagw} & \xleftarrow{[\text{AKK}^+15]} & \text{kw} & \xleftarrow{[\text{GHK}^+14]} & \text{d-pw} & \xleftarrow{[\text{Gru08}]} & \text{cr} & \xleftarrow{[\text{GHK}^+14]} & \text{fvs} & \longleftarrow & \text{fas} \end{array}$$

Table 3.5: Known relations between digraph width measures. A directed edge from measure β to measure α indicates that within the cited work there has been shown some function f such that for every digraph it holds $\alpha(G) \leq f(\beta(G))$.

As seen in this section, it is not possible to compare non-linear directed width parameters as good as linear directed width parameters. Further, they are all at least NP-hard, DAG-width even PSCPACE-hard to compute. It therefore seems likely to consider width parameters on restricted graph classes, as we do in the following sections.

3.5.4 Coloring and directed width parameters

As mentioned before, graph coloring can be understood as a directed parameter by definition, but is not one of the standard width parameters. Even though, it is an interesting matter to investigate the graph coloring problem on classes of bounded width.

We will here only shortly summarize the results of [GKR21a].

Corollary 3.5.26. *The Dichromatic Number problem is W[1]-hard on complete bioriented digraphs and thus, on all digraphs when parameterized by directed clique-width.*

Theorem 3.5.27. *The Dichromatic Number problem on digraphs on n vertices given by a directed clique-width k-expression can be solved in $n^{2^{\mathcal{O}(k^2)}}$ time.*

Corollary 3.5.28. *The Dichromatic Number problem is in XP when parameterized by directed clique-width.*

4 Directed Graph Minors

Graph minors are an important tool to characterize directed graph classes. Several special graph classes can be defined by a set of forbidden graph minors. Further, in the undirected case, by Halins grid theorem, there is a strong relationship between the tree-width of a graph and the size of its largest minor grid, shown by Robertson et al. in [RS86b, RST94]. For directed graphs, Kreutzer et al. could show a relationship between the directed tree-width of a graph and the size of its largest cylindrical minor grid [KK15, HKK19].

Therefore, concerning directed graph parameters, it seems likely to consider directed versions of graph minors.

Please note that the definitions in this sections can be refound in [GR19b].

Definition 4.0.1 (Directed edge contraction)**.** Let $G = (V, E)$ be a digraph with $e = (u, v) \in E$. The *contraction* of e leads to a new digraph $G' = (V', E')$ with $V' = V \setminus \{u, v\} \cup \{w\}$ with $w \notin V$ and $E' = \{(a, b) \mid a, b \in V \cap V',\ (a, b) \in E$ or $a = w, (u, b)$ or $(v, b) \in E$ or $b = w, (a, u)$ or $(a, v) \in E\}$.[1]

There are different ways of defining graph minors using directed edge contraction. As directed path-width and directed tree-width are not monotone under the directed edge contraction on every edge, it is sensible to restrict the edges, on which directed edge contraction can be used. We introduce an equivalent definition to the one introduced by Kintali and Zhang in [KZ17]. Therefore we need to define cycle contraction:

Definition 4.0.2 (Directed cycle contraction)**.** Let $G = (V, E)$ be a digraph with $C = \{v_1, \ldots, v_\ell\}$ a cycle. The *contraction* of C leads to a new digraph $G' = (V', E')$ with $V' = V \setminus C \cup \{w\}$ with $w \notin V$ and $E' = \{(a, b) \mid a, b \in V \cap V',\ (a, b) \in E$ or $a = w, (v_i, b) \in E$ for $1 \leq i \leq \ell$ or $b = w, (a, v_i) \in E$ for $1 \leq i \leq \ell\}$.[2]

Butterfly contractions are defined by Johnson et al. in [JRST01b] as directed edge contractions of an edge $e = (u, v)$, where either e is the only outgoing edge of u or e is the only incoming edge of v. The definition of out-contraction of [KZ17] is

[1] This means, in digraph G' the edge e and its two incident vertices u and v are replaced by the vertex w and all other edges in G incident with u or v are incident with w in G'.

[2] This means, in digraph G' the cycle C is replaced by the vertex w and all other edges in G incident with a vertex in C are incident with w in G'.

equal to deleting all outgoing edges of u but e and then doing a butterfly contraction, the definition of in-contraction is equal to deleting all incoming edges of v but e and doing a butterfly contraction of e. Therefore, the following definition of directed graph minors is equal to the one given in [KZ17]:

Definition 4.0.3 (Directed graph minor)**.** Let $G = (V, E)$ be a digraph. A digraph $G' = (V', E')$ is a *directed minor* of G, i.e. $G' \preceq G$, if G' can be obtained by creating subgraphs, performing cycle contractions and performing butterfly contractions on G.

Furthermore, the directed graph minor relation is transitive, reflexive and anti-symmetric, but not symmetric.

Note that unfortunately, unlike assumed by Johnson et al. in [JRST01b], directed tree-width is not closed under the butterfly minor operation. This has been shown by a counterexample in [Adl07]. Therefore, it seems impossible to get something like the grid theorem on directed tree-width. However, it seems to be true that directed tree-width does not increase a lot by the butterfly minor operation. Further, it seems possible to define the class of digraphs with directed tree-width 1 by forbidden minors, using hypergraphs. In [Wie20], Wiederrecht gives some characterizations with directed graph minors.

5 Width Measures on Directed Graph Classes

5.1 Tree-Like Digraphs

In this section we consider directed version of cactus trees and forests as well as pseudotrees and -forests, which are digraphs resembling to trees. We show that these graph classes are characterizable by directed graph minors and show that they have bounded directed tree-width or even path-width. Please note that huge parts of this section are taken from [GR19b].

5.1.1 Directed Cactus Forests and Pseudoforests

First we will apply the definitions of undirected cactus trees and forests as well as pseudotrees and -forests to directed graphs. For directed cactus trees, it is possible to use nearly the same definition as for undirected cactus trees:

Definition 5.1.1 (Directed cactus tree). A *directed cactus tree* is a strongly connected digraph $G = (V, E)$, where for any two directed cycles C_1 and C_2 it holds that they have at most one joint vertex.

This definition remains equal if C_1 and C_2 must have exactly one joint vertex, and it is equal to the definition given in [BJG18]:

Definition 5.1.2 (Directed cactus tree). A *directed cactus tree* is a strongly connected digraph in which each arc is contained in exactly one directed cycle. The class of all directed cactus trees is named DCT.

It would also be possible to define cactus trees as weakly connected subgraphs, where two directed cycles have at most one joint vertex. This would lead to a superset of Definition 5.1.1 and a subset of directed cactus forests, which can be defined as follows:

Definition 5.1.3 (Directed cactus forest). A *directed cactus forest* is a digraph $G = (V, E)$, where for any two directed cycles C_1 and C_2 it holds that they have at most one joint vertex. The class of all directed cactus forests is named DCF.

Note that if G does not need to be strongly connected, it is not equal if C_1 and C_2 have exactly one directed cycle. It though holds that a graph is a directed pseudoforest, if and only if each arc is contained in at most one cycle. It further holds that if G is a directed cactus tree, then its underlying (undirected) graph $und(G)$ is a cactus tree. But if G is a directed cactus forest, the underlying graph does not need to be neither a cactus tree nor a graph of which every connected component is a cactus tree. The other way around is only true if we use an orientation where no bioriented arcs are allowed. Then if G is an undirected cactus tree or a graph of which every connected component is a cactus tree, then every orientation of G is a directed cactus forest.

For pseudotrees, there are also different ideas of defining a directed version, depending on whether strong or weak connectivity is used. Here it is more sensible to use weak connectivity, because a strongly connected graph containing at most one cycle is exactly a cycle.

Definition 5.1.4 (Directed pseudotree)**.** A *directed pseudotree* is a weakly connected digraph which contains at most one directed cycle. The class of all directed pseudotrees is named DPT.

In contrast to directed cactus forests, it does matter for directed pseudoforests if we consider strong or weak connectivity:

Definition 5.1.5 (Directed weak pseudoforest)**.** A *directed weak pseudoforest* is a digraph, in which every weakly connected component contains at most one directed cycle. The class of all directed weak pseudoforests is named DWPF.

Definition 5.1.6 (Directed strong pseudoforest)**.** A *directed strong pseudoforest* is a digraph, in which every strongly connected component contains at most one directed cycle, i.e. contains exactly one directed cycle. The class of all directed strong pseudoforests is named DSPF.

Then directed strong pseudoforests are a superclass of directed weak pseudoforest, as every strongly connected component is also a weakly connected component. It further holds, that directed strong pseudoforests are exactly those graphs, where any two directed cycles have no joint vertex, or where every vertex is in at most one cycle.

Note that here as well it holds that if G is a directed pseudotree, the underlying graph $und(G)$ is an undirected pseudotree. For directed weak pseudoforests the underlying undirected graphs are undirected pseudoforests, but for directed strong pseudoforests this is not generally true. But it holds that every orientation of an undirected pseudoforest, is a directed strong pseudoforest.

Proposition 5.1.7. *We have the following inclusions for tree-like digraphs.*

$$DPT \subset DCT \subset DCF \tag{5.1}$$

$$DPT \subset DWPF \subset DSPF \subset DCF \tag{5.2}$$

5.1.2 Directed Graph Minors of Tree-like Digraphs

As cactus forests and pseudoforests are characterizable by forbidden graph minors, we want to characterize their directed versions by forbidden directed graph minors.

Directed pseudotrees and directed cactus trees can not be closed under directed minor operations, as they are not even closed under the subgraph operation. Directed cactus forests and directed strong/weak pseudoforests are closed under directed graph minor operations by the following results.

Lemma 5.1.8. *Directed cactus forests are closed under directed graph minor operations.*

Proof. Let $G = (V, E)$ be a directed cactus forest. Then it holds for every two cycles C_1, C_2 that they have at most one joint vertex. That is, for all $e \in E$ holds that e is part of at most one cycle.

- *Subdigraphs:* By deleting vertices or arcs, no edge can become part of another cycle.
- *Butterfly contraction:* Let $e = (u, v)$ be an arc in G such that e is the only outgoing edge of u or e is the only incoming edge of v. Then there is no path from u to v in $G-(u, v)$. Then no additional cycle can be created by contraction of e, as no additional arc is created and therefore the only additional possibility to create a new cycle would be containing the new vertex w and a path from w to w, in G, which has not been a path from v to u in G. This is a contradiction to that there is no path from u to v in G. It follows that every arc is still only in at most one cycle in G'.
- *Cycle contraction* Let C be a cycle in G. By contracting C, no new cycle can be created, as no additional arc is created and there has already been a path from u to v and from v to u for all $u, v \in C$. Therefore, assigning C to only one vertex w does not create new path from w to w. Thus, every arc is still only in at most one cycle in G'.

□

Lemma 5.1.9. *Directed strong/weak pseudoforests are closed under directed graph minor operations.*

Proof. We use the same argument as in Lemma 5.1.8. For subgraphs, by deleting vertices or arcs, no edge can become part of another cycle. By butterfly and cycle contraction, no additional cycles can be created. From this also follows that these contractions can not create additional strongly connected components. It is easy to see that both contractions can not create weakly connected components, as only arcs are considered, and there are no arcs between two weakly connected components. As directed strong/weak pseudoforests are defined as graphs, where each strong/weakly connected component contains at most one cycle, this means that directed strong/weak pseudoforests are closed under directed graph minor operations. □

So directed cactus forests and directed strong and weak pseudoforests are closed under graph minor operation. But even more, it is possible to characterize those classes by a finite number of forbidden directed graph minors:

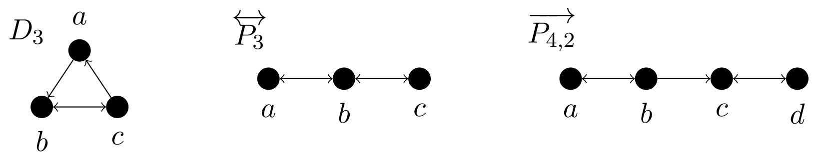


Figure 5.1: The forbidden directed minors D_3, $\overleftrightarrow{P_3}$ and $\overrightarrow{P_{4,2}}$.

Theorem 5.1.10. *Digraph G is a directed cactus forest if and only if it does not contain the digraph D_3, the directed cycle $\overrightarrow{C_3}$ with one additional arc, as a directed graph minor.*

Proof. $\subseteq$ Let G be a directed cactus forest. Assume that D_3 is a minor of G. As there are two cycles in D_3, $C_1 = \{a, b, c\}$ and $C_2 = \{b, c\}$ containing the vertex b as well as the vertex c, D_3 is not a directed cactus forest. Then Lemma 5.1.8, leads to a contradiction.

$\supseteq$ Let G be a digraph with no D_3 as a directed minor. Assume, that G is not a cactus graph. Then there is an arc $e = (u, v)$ in G, such that there are two cycles C_1, C_2 with $u, v \in C_1$ and $u, v \in C_2$. By subgraph operations we obtain G' which contains only of C_1 and C_2 as a graph minor of G. Using then butterfly minor operations on all arcs of C_2 but e and on all arcs of C_1 but e and two other arcs, we obtain D_3 as a directed minor of G'. Then D_3 is a directed minor of G, which leads to a contradiction.

□

Further, it holds that D_3 is the minimal forbidden minor for directed cactus forests, as every further minor operation would lead to a graph with only one cycle, so every graph minor of D_3 is a directed cactus forest.

Theorem 5.1.11. *Digraph G is a directed strong pseudoforest if and only if it does not contain the digraph D_3 or the digraph $\overleftrightarrow{P_3}$ as a directed graph minor.*

Proof. $\subseteq$ Let G be a directed strong pseudoforest. Assume that D_3 or $\overleftrightarrow{P_3}$ is a minor of G. Both D_3 and $\overleftrightarrow{P_3}$ consist of only one strongly connected component, but include two cycles $\{a, b, c\}$ and $\{b, c\}$ for D_3 and $\{a, b\}$ and $\{b, c\}$ for $\overleftrightarrow{P_3}$, both graphs are no directed strong pseudoforest. Thus Lemma 5.1.9, leads to a contradiction.

⊇ Let G be a digraph with no D_3 or $\overleftrightarrow{P_3}$ as directed minor. Assume that G is not a directed strong pseudoforest. Then G includes a strongly connected component, which has at least two cycles. Let G' be the subgraph of G which only consists of this strongly connected component.

Case 1 Assume that the two cycles in G' have a joint arc. Then, as in the proof of Theorem 5.1.10, G' and therefore G has D_3 as a directed minor. This is a contradiction.

Case 2 Assume that the two cycles in G' do not join an arc. As G' is strongly connected, there are two cycles C_1 and C_2 in G' which have a joint vertex. By subgraph operations, delete all arcs and vertices except these two cycles. Then use butterfly contractions to transform these cycles to cycles of size 2. By this, $\overleftrightarrow{P_3}$ results as a directed minor of G. This leads to a contradiction.

□

Theorem 5.1.12. *Digraph G is a directed weak pseudoforest if and only if it does not contain the digraph D_3, the digraph $\overleftrightarrow{P_3}$ or the digraph $\overrightarrow{P_{4,2}}$ as a directed graph minor.*

Proof. ⊆ Let G be a directed weak pseudoforest. Assume that D_3, the graph $\overleftrightarrow{P_3}$ or the digraph $\overrightarrow{P_{4,2}}$ is a minor of G. As all three graphs contain only one weakly connected component, but two cycles, they are no directed strong pseudoforests. Then Lemma 5.1.9 leads to a contradiction.

⊇ Let G be a digraph with no D_3, $\overleftrightarrow{P_3}$ or $\overrightarrow{P_{4,2}}$ as directed minor. Assume that G is not a directed weak pseudoforest. Then G includes a weakly connected component, which has at least two cycles. Let G' be the subgraph of G which only consists of this weakly connected component.

Case 1 Assume that any two cycles in G' have a joint arc. Then, as in the proof of Theorem 5.1.10, G' and therefore G has D_3 as a directed minor. This is a contradiction.

Case 2 Assume that all two cycles in G' do not join an arc, but there are two cycles which have a joint vertex. Then, as in the proof of Theorem 5.1.11, G' and therefore G has $\overleftrightarrow{P_3}$ as a directed minor. This is a contradiction.

Case 3 Assume that any two cycles in G' do not have a joint vertex. Let C_1, C_2 be two cycles in G'. By subgraph operations, delete all arcs and vertices except C_1 and C_2 and a directed path connecting C_1 and C_2. Then use butterfly contractions to transform C_1 and C_2 to cycles of size 2 and the path connecting them to a path of length 1. By this, $\overrightarrow{P_{4,2}}$ results as a directed minor of G. This is a contradiction.

□

5.1.3 Directed Path-Width of Tree-like Digraphs

In order to process the strong components of a digraph we recall the following definition. The *acyclic condensation* of a digraph G, $AC(G)$ for short, is the digraph whose vertices are the strongly connected components $V_1, \ldots, V_c$ of G and there is an edge from V_i to V_j if there is an edge (v_i, v_j) in G such that $v_i \in V_i$ and $v_j \in V_j$. Obviously for every digraph G the digraph $AC(G)$ is always acyclic.

Let $G_1 = (V_1, E_1), \ldots, G_k = (V_k, E_k)$ be k vertex-disjoint digraphs. The *directed union* of $G_1, \ldots, G_k$, denoted by $G_1 \ominus \ldots \ominus G_k$, is defined by their disjoint union plus possible arcs from vertices of G_i to vertices of G_j for all $1 \leq i < j \leq k$.

Theorem 5.1.13. *Directed cactus trees have unbounded directed path-width.*

Proof. Let G be the complete biorientation of the undirected, binary tree of height h. We know that the path-width of perfect binary trees of hight h is $\lceil h/2 \rceil$ (cf. [Sch89]). Then, by Lemma 3.2.4 it follows that $\text{d-pw}(G) = \text{pw}(und(G)) = \lceil h/2 \rceil$. As all complete biorientations of binary trees are directed cactus trees, it follows that directed path-width is not bounded for directed cactus trees. □

As all directed cactus trees are directed cactus forests, it follows directly:

Corollary 5.1.14. *Directed cactus forests have unbounded directed path-width.*

This is not true for directed strong or weak pseudoforests. As complete biorientations of binary trees are no directed pseudoforests, neither strong or weak, as they consist of only one strongly connected component, but contain lots of cycles, the counterexample from the proof of Theorem 5.1.13 does not work here. Further, it holds that this graph class has bounded directed path-width:

Theorem 5.1.15. *Directed strong pseudoforests have directed path-width at most* 1.

Proof. Let $G = (V, E)$ be a directed strong pseudoforest. Every strong component has at least size one, so the smallest strong components could be single vertices. Let C be a strongly connected component of G. As G is a pseudoforest, C is exactly a directed cycle. For every directed cycle $C = \{c_1, \ldots, c_r\}$ with arcs (c_i, c_{i+1}) for $1 \leq i \leq r-1$ and (c_r, c_1) we give a directed path-decomposition as follows: For the cycle with $r = 1$ vertex a path-decomposition consists of only one bag, which only contains this single vertex. This is obviously a directed path-decomposition of width 0. For cycles with $r > 1$ vertices, we construct $X_1, \ldots, X_{r-1}$ with $X_1 = \{c_1, c_2\}$, $X_2 = \{c_1, c_3\}$, $\ldots$, $X_{r-1} = \{c_1, c_r\}$. Then $\mathcal{X} = (X_1, \ldots, X_{r-1})$ is a directed path-decomposition of C of width 1. As each strong component of G has directed path-width at most 1, by Lemma 3.2.8 the digraph G also has directed path-width at most 1.

□

Since the proof of Lemma 3.2.8 using the results of [GR18] is constructive, we even can give a directed path-decomposition of width 1 for every (not strongly connected) directed pseudoforest. As directed strong pseudoforests are a superclass of weak pseudoforests and directed weak pseudoforests are a superclass of directed pseudotrees, it follows directly:

Corollary 5.1.16. *Directed weak pseudoforests and directed pseudotrees have directed path-width at most* 1.

5.1.4 Directed Tree-Width of Tree-like Digraphs

Theorem 5.1.17. *Directed cactus forest have directed tree-width at most* 1.

Remark 5.1.18. Every strong component of a directed cactus forest G consists of r cycles $C_1, \ldots, C_r$ such that for every C_i, $1 \leq i \leq r$, there is a C_j with $i \neq j$, $1 \leq j \leq r$ such that C_i and C_j have exactly one joint vertex. Further, there is a C_i, $1 \leq i \leq r$ such that there is exactly one other cycle C_j with $i \neq j$, $1 \leq j \leq r$ such that C_i and C_j have exactly one joint vertex.

Proof of Theorem 5.1.17. Let G be a directed cactus forest. By Lemma 3.3.11, the directed tree-width of G is the maximum directed tree-width of the strong components of G. So we only need to consider the strong components of G. Let G' be a strong component of G. By Remark 5.1.18, G' consists of r cycles $C_1, \ldots, C_r$ and there is a C_i, $1 \leq i \leq r$ such that there is exactly one other cycle C_j with $i \neq j$, $1 \leq j \leq r$ such that C_i and C_j have exactly one joint vertex. To give a directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ for the strong component of G, we start with a vertex of this C_i. A directed tree decomposition of a cycle $C_i = \{c_{i,1}, \ldots, c_{i,\ell}\}$ is always given by a path T and bags $W_{i,t} = \{c_{i,t}\}$ for all $1 \leq t \leq \ell$ and edge sets $X_{(c_{i,t}, c_{i,t+1})} = \{c_{i,t}\}$. Since the order of the vertices in C_i is not unique, our construction leads to a directed tree-decomposition for any order of the vertices in C_i. So we can start with any vertex in C_i and create a directed tree-decomposition for this cycle.

By Remark 5.1.18, there is at least one cycle C_j, $i \neq j$, $1 \leq j \leq r$ which has a joint vertex with C_i. So for $C_j = \{c_{j,1}, \ldots, c_{j,k}\}$ there is some $c_{i,q}$, $1 \leq q \leq \ell$ such that $c_{j,1} = c_{i,q}$. (Without loss of generality order C_j in a way such that $c_{j,1}$ is the joint vertex with C_i.) Then append the vertices of C_j to the directed tree-decomposition by creating new bags $W_{j,t} = \{c_{j,t}\}$ for all $1 < t \leq k$ and edges $X_{(c_{j,t}, c_{j,r})} = \{c_{i,t}\}$ for $2 \leq i < k$, $2 < r \leq k$ and $X_{(c_{j,1}, c_{j,2})} = X_{(c_{i,q}, c_{j,2})} = \{c_{i,q}\} = \{c_{j,1}\}$.

By Remark 5.1.18 and as of course the strong components of G are strongly connected, there is always a next cycle to insert in the same way somewhere in the tree structure T of our tree-decomposition, till all vertices of the strong component are in a bag of the directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$. It remains to show that $(T, \mathcal{X}, \mathcal{W})$ really is a directed tree-decomposition of width 1 for a strong component of G:

(dtw-1) $\mathcal{W} = \{W_r \mid r \in V_T\}$ is a partition of V_G into nonempty subsets. As already said, all vertices of G are inserted one by one in bags W by the fact that they are all strongly connected and share a vertex with another cycle. Further, no vertex occurs twice, as in a cactus forest all cycles share at most one joint vertex, and this joint vertex is not added a second time in a W-set.

(dtw-2) For every $(u, v) \in E_T$ the set $\bigcup\{W_r \mid r \in V_T, v \leq r\}$ is $X_{(u,v)}$-normal. Let $(u, v) \in E_T$. Then it holds, by the definition of T that there is a cycle

C_j in G such that $(u, v) = (c_{j,t}, c_{j,t+1})$ for $c_{j,t}, c_{j,t+1}$ are elements of the cycle $C_j = (C_{j,1}, \ldots, C_{j,k})$. Further, it holds that $X_{(u,v)} = \{c_{j,t}\}$. By the definition of $(T, \mathcal{X}, \mathcal{W})$ the set $\bigcup\{W_r \mid r \in V_T, v \leq r\}$ consists of a number of cycles, lets say $C_{j+1}, \ldots, C_r$ and the vertices $\{c_{j,t+1}, \ldots, c_{j,k}\}$. As any to cycles in G have at most one vertex in common, it is not possible that there is an arc from one of those cycles to one of the cycles in $C_1, \ldots, C_{j-1}$, as this would create a big cycle including lots of vertices and edges from the cycles this arc would connect. So the only way to get a path from $\bigcup\{W_r \mid r \in V_T, v \leq r\}$ out and back in this set is by using the cycle C_j. It follows that in $G' - X_{(u,v)} = G' - \{c_{j,t}\}$ there is no path out and back in the set $\bigcup\{W_r \mid r \in V_T, v \leq r\}$, which means that this set is $X_{(u,v)}$-normal.

It further holds that $W_r \cup \bigcup_{e \sim r} X_e = \{c_{j,t}\} \cup \{c_{j,t}\} \cup \{c_{j,t-1}\} = \{c_{j,t}, c_{j,t+1}\}$ for all W_r for some C_j cycle of G' and $t > 1$. It then follows that $\max_{r \in V_T} |W_r \cup \bigcup_{e \sim r} X_e| - 1 = 2 - 1 = 1$, so the directed tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of G' has width 1. It therefore follows that each strong component of G has directed tree-width at most 1, so G has directed tree-width at most 1. □

Since the proof of Lemma 3.3.11 using the results of [GR18] is constructive, we even can give a directed tree-decomposition of width 1 for every (not strongly connected) directed cactus forest.

As directed pseudoforests and directed cactus trees are both subclasses of directed cactus forests, we can conclude the following corollaries. The first statement also follows by Theorem 5.1.15, as the directed tree-width of a graph is always smaller or equal to the directed path-width of this graph [GR18].

Corollary 5.1.19. *Directed strong/weak pseudoforests have directed tree-width at most* 1.

Corollary 5.1.20. *Directed cactus trees have directed path-width at most* 1.

The other direction of Theorem 5.1.17 does not hold true. There are graphs of directed tree-width 1 which are not directed cactus forests, as for example their forbidden directed graph minor D_3. This graph has directed path-width 1 by the directed path-decomposition $\mathcal{X} = (X_1, X_2)$ with $X_1 = \{a, c\}$ and $X_2 = \{b, c\}$. It then follows that it also has directed tree-width at most 1 and as it includes a cycle, it has directed tree-width exactly 1.

5.1.5 Conclusion and Outlook

We now introduced directed cactus trees (DCT) and forests (DCF), directed pseudotrees (DPT) and directed strong (DSPF) and weak pseudoforests (DWPF). We could prove that DCF, DSPF, and DWPF can be characterized by at most three forbidden digraph minors, using a graph minor operation for which directed path-width is monotone. Furthermore, we showed that DCF and its subclasses have directed tree-width at most 1 and DSPF, DCT and their subclasses even have directed path-width at most 1.

We also considered an oriented version of Halin graphs by connecting the leaves within a planar embedding of an out-tree in their clockwise ordering. This leads to a subclass of DWPF as well as DAGs. But these graphs can not be closed under directed minor operations, as they are not even closed under the subgraph operation.

In the paper [GR19b], we wrote that our results should be a first step on the way to find forbidden directed graph minors for the classes of directed tree-width at most 1 and classes of directed path-width at most 1. The latter have already been proven to have a countable number of forbidden directed graph minors [KZ15], but these minors could not be found yet. Finding them could be an issue of future work, as well as checking if there is a countable number of forbidden digraph minors for the set of digraphs of directed tree-width at most 1 and to find them.

Note that since then there has been some further works to characterize the class of digraphs oft directed tree-widht at most 1 in [Wie20]. There, this matter could be solved using hypergraphs.

5.2 Directed Co-Graphs

As undirected co-graphs have already been introduced in the 1970s, there has been a lot of research on this graph class right now. Regarding graph parameters, a very important result has been presented by Bodlaender and Möhring in [BM90, BM93]. There, the authors show that tree-width and path-width are computable in linear time on co-graphs and further, both parameters are equal for each co-graph.

This leads to the question, if that result is expandable to a directed version of co-graphs. As already mentioned before, there has been many attempts to give directed versions of path-width and tree-width. We will therefore not only consider directed path-width and directed tree-width, but also other directed graph parameters. We consider the parameters directed path-width(d-pw), directed tree-width (d-tw), directed feedback vertex set number (fvs), directed feedback arc set number (fas), cycle rank (cr), DAG-depth (ddp), DAG-width (dagw) and Kelly-width (kw). As also mentioned before, the minimization problem for these parameters is generally NP-hard. We will now show that this is not true on directed co-graphs.

We therefore show useful properties of the width measures decompositions. The bidirectional complete subdigraph and bidirectional complete bipartite subdigraph lemmas give tight connections between such subdigraphs and the bags in decompositions of the considered digraph. These properties allow us to show how the measures can be computed for the disjoint union, order composition, and series composition of two directed graphs. Our proofs are constructive, i.e. a decomposition can be computed from a given co-expression.

Our results imply that for every directed co-graph G, we have

$$\mathrm{kw}(G)-1 \leq \text{d-pw}(G) = \text{d-tw}(G) = \mathrm{cr}(G) = \mathrm{dagw}(G)-1 \leq \mathrm{fvs}(G) \leq \mathrm{fas}(G) \quad (5.3)$$

and

$$\mathrm{dagw}(G) \leq \mathrm{ddp}(G). \quad (5.4)$$

We thereby give linear time solutions to compute these width parameters for the disjoint union, series composition and for the order composition of two directed graphs. This leads to a constructive linear-time-algorithm for computing the width as well as the according decompositions of a directed co-graph. This works for all parameters in (5.3) and (5.4) except for Kelly-width, which does not allow a simple method for the series operation of two digraphs. Furthermore, we obtain that for directed co-graphs Kelly-width can be bounded by DAG-width (Theorem 5.2.34). Due to [HK08, Conjecture 30], [AKK$^+$15], and [BJG18, Section 9.2.5] this question remains open for general digraphs.

For most of the parameters, we could even expand the algorithms to extended directed co-graphs, which are an extension of the directed co-graphs defined in [CP06] by an additional transformation considered in [JRST01b].

We thus prove that many directed width measures are linearly computable on the special class of directed co-graphs. As directed co-graphs have many applications, this is very helpful to use FPT-algorithms with those directed width measures as parameter in these cases.

This section presents the results of [GR18, GKR19a, GKR21b]. Huge parts of the following section are taken from [GKR21b].

5.2.1 Recursively defined Digraphs

Operations and Transformations

Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two vertex-disjoint digraphs. The following operations have already been considered by Bechet et al. in [BdGR97].

- The *disjoint union* of G_1 and G_2, denoted by $G_1 \oplus G_2$, is the digraph with vertex set $V_1 \cup V_2$ and arc set $E_1 \cup E_2$.

- The *series composition* of G_1 and G_2, denoted by $G_1 \otimes G_2$, is defined by their disjoint union plus all possible arcs between vertices of G_1 and G_2.

- The *order composition* of G_1 and G_2, denoted by $G_1 \oslash G_2$, is defined by their disjoint union plus all possible arcs from vertices of G_1 to vertices of G_2.

The following transformation has already been considered by Johnson et al. in [JRST01b] and generalizes the operations disjoint union and order composition.

- A graph G is obtained by a *directed union* of G_1 and G_2, denoted by $G_1 \ominus G_2$, if G is a subdigraph of the order composition of $G_1 \oslash G_2$ and contains the disjoint union $G_1 \oplus G_2$ as a subdigraph.

Please note that the directed union is not unique and thus no operation.

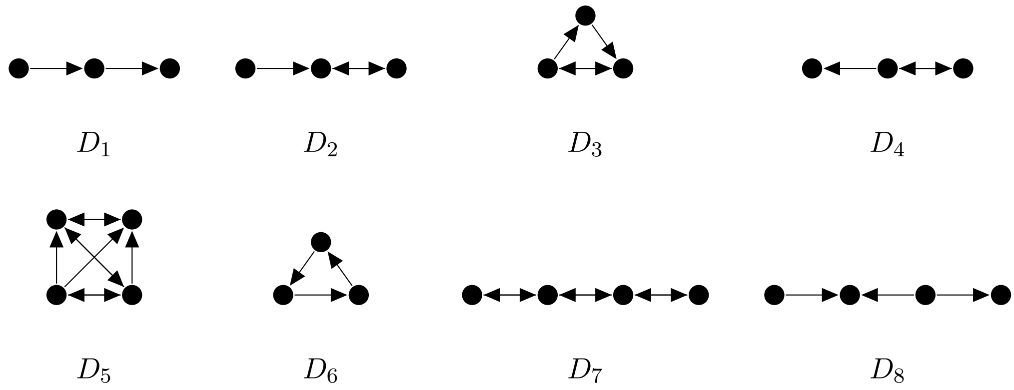


Figure 5.2: The forbidden subdigraphs for directed co-graphs

Directed co-graphs

We recall the definition of directed co-graphs from [CP06].

Definition 5.2.1 (Directed co-graphs, [CP06])**.** The class of *directed co-graphs* is recursively defined as follows.

(i) Every digraph on a single vertex $(\{v\}, \emptyset)$, denoted by $\bullet$, is a *directed co-graph*.

(ii) If G_1 and G_2 are vertex-disjoint directed co-graphs, then

  (a) the disjoint union $G_1 \oplus G_2$,

  (b) the series composition $G_1 \otimes G_2$, and

  (c) the order composition $G_1 \oslash G_2$ are *directed co-graphs*.

By this definition we conclude that for every directed co-graph $G = (V, E)$ the underlying undirected graph $und(G)$ is a co-graph. On the other hand, not every orientation of an undirected co-graph is a directed co-graph.

In [CP06] it has been shown that directed co-graphs can be characterized by eight forbidden induced subdigraphs. Those digraphs are shown in Figure 5.2.

Obviously, for every directed co-graph we can define a binary tree structure, denoted as *di-co-tree*. The leaves of the di-co-tree represent the vertices of the graph and the inner nodes of the di-co-tree correspond to the operations applied on the subexpressions defined by the subtrees. For every directed co-graph one can construct a di-co-tree in linear time, see [CP06].

Using the di-co-tree a lot of hard problems have been shown to be solvable in polynomial time when restricted to directed co-graphs [Gur17].

Lemma 5.2.2. *For a digraph G the following properties hold.*

1. *Digraph G is a directed co-graph if and only if digraph $\overline{G}$ is a directed co-graph.*

2. *Digraph G is a directed co-graph if and only if digraph G^c is a directed co-graph.*

Proof. We sketch the proof by a simple modification of the directed co-graph expression of G.

1. Let X be a directed co-graph expression for digraph G. We can get $\overline{G}$ given by an expression X' by modifying X as follows. Every directed union in X is a series composition in X' and every series composition in X is a disjoint union in X'. Further if $X_1 \oslash X_2$ in X, we change this into $X_2 \oslash X_1$ in X'.

2. Let X be a directed co-graph expression for digraph G. We can get G^c given by an expression X' by modifying X as follows. For every $X_1 \oslash X_2$ in X, we change this into $X_2 \oslash X_1$ in X'. The rest remains as in X.

□

Extended directed co-graphs

Since the directed union generalizes the disjoint union and also the order composition, we can generalize the class of directed co-graphs as follows.

Definition 5.2.3 (Extended directed co-graphs)**.** The class of *extended directed co-graphs* is recursively defined as follows.

(i) Every digraph on a single vertex $(\{v\}, \emptyset)$, denoted by $\bullet$, is an *extended directed co-graph.*

(ii) If G_1 and G_2 are vertex-disjoint extended directed co-graphs, then

   (a) every directed union $G_1 \ominus G_2$ and

   (b) the series composition $G_1 \otimes G_2$ are *extended directed co-graphs.*

Also for every extended directed co-graph we can define a tree structure, denoted as *ex-di-co-tree.* The leaves of the ex-di-co-tree represent the vertices of the graph and the inner nodes of the ex-di-co-tree correspond to the operations applied on the subexpressions defined by the subtrees. For the class of extended directed co-graphs it remains open how to compute an ex-di-co-tree.

By applying the directed union, which is not a disjoint union and an order composition, we can obtain digraphs whose complement digraph is not an extended directed co-graph. An example for this leads the directed path on 3 vertices $\overrightarrow{P_3}$. Thus, we only can carry over one of the two results shown in Lemma 5.2.2 to the class of extended directed co-graphs.

Lemma 5.2.4. *Let G be some digraph. Digraph G is an extended directed co-graph if and only if digraph G^c is an extended directed co-graph.*

Proof. Let X be an expression using extended directed co-graph operations for G. We can get G^c given by an expression X' by modifying X as follows. For every $X_1 \ominus X_2$ in X, we change this into $X_2 \ominus X_1$ in X'. The rest remains as in X. □

Oriented Co-Graphs

Oriented colorings are defined on oriented graphs, which are digraphs with no bidirected edges. Therefore we introduce oriented co-graphs by omitting the series operation from the definition of directed co-graphs, as given in [BdGR97]

Definition 5.2.5 (Oriented Co-Graphs)**.** The class of *oriented co-graphs* is recursively defined as follows.

1. Every digraph on a single vertex $(\{v\}, \emptyset)$, denoted by $\bullet$, is an *oriented co-graph*.
2. If G_1, G_2 are k vertex-disjoint oriented co-graphs, then
   (a) $G_1 \oplus G_2$ and
   (b) $G_1 \oslash G_2$ are *oriented co-graphs*.

The class of oriented co-graphs was already analyzed by Lawler in [Law76] and [CLSB81, Section 5] using the notation of *transitive series parallel (TSP) digraphs*. A digraph $G = (V, A)$ is called *transitive*, if for every pair $(u, v) \in A$ and $(v, w) \in A$ of arcs with $u \neq w$ the arc (u, w) also belongs to A.

Theorem 5.2.6 ([CLSB81])**.** *A graph G is a co-graph if and only if there exists an orientation G' of G, such that G' is an oriented co-graph.*

A di-co-tree T of an oriented co-graph is *canonical* if on every path from the root to the leaves of T, the labels disjoint union and order operation strictly alternate. Since the disjoint union $\oplus$ and the order composition $\oslash$ are associative, we always can assume canonical di-co-trees.

Lemma 5.2.7. *Let G be an oriented co-graph and T be a di-co-tree for G. Then, T can be transformed in linear time into a canonical di-co-tree for G.*

The recursive definitions of oriented and undirected co-graphs lead to the following observation.

Observation 5.2.8. For every oriented co-graph G the underlying undirected graph $und(G)$ is a co-graph.

The reverse direction of this observation only holds under certain conditions, see Theorem 5.2.10. By $\overleftrightarrow{P_2} = (\{v_1, v_2\}, \{(v_1, v_2), (v_2, v_1)\})$ we denote the complete biorientation of a path on two vertices.

Lemma 5.2.9. *Let G be a digraph, such that $G \in Free(\{\overleftrightarrow{P_2}, D_1, D_6\})$. Then, it holds that G is transitive.*

Proof. Let $(u, v), (v, w) \in A$ be two arcs of $G = (V, A)$. Since $G \in \text{Free}(\{\overleftrightarrow{P_2}\})$, we know that $(v, u), (w, v) \notin A$. Further, since $G \in \text{Free}(\{D_1, D_6\})$, we know that u and w are connected either only by $(u, w) \in A$ or by $(u, w) \in A$ and $(w, u) \in A$, which implies that G is transitive. $\square$

Oriented co-graphs can be characterized by forbidden subdigraphs as follows.

Theorem 5.2.10. *Let G be a digraph. The following properties are equivalent:*

1. *G is an oriented co-graph.*
2. *$G \in Free(\{D_1, D_6, D_8, \overleftrightarrow{P_2}\})$.*
3. *$G \in Free(\{D_1, D_6, \overleftrightarrow{P_2}\})$ and $und(G) \in Free(\{P_4\})$.*
4. *$G \in Free(\{D_1, D_6, \overleftrightarrow{P_2}\})$ and $und(G)$ is a co-graph.*
5. *G has directed NLC-width 1 and $G \in Free(\{\overleftrightarrow{P_2}\})$.*
6. *G has directed clique-width at most 2 and $G \in Free(\{\overleftrightarrow{P_2}\})$.*
7. *G is transitive and $G \in Free(\{\overleftrightarrow{P_2}, D_8\})$.*

Proof. (1) $\Rightarrow$ (2) If G is an oriented co-graph, then G is a directed co-graph and by [CP06] it holds that $G \in \text{Free}(\{D_1, \ldots, D_8\})$. Furthermore, $G \in \text{Free}(\{\overleftrightarrow{P_2}\})$ because of the missing series composition. This leads to $G \in \text{Free}(\{D_1, D_6, D_8, \overleftrightarrow{P_2}\})$.(2) $\Rightarrow$ (1) If $G \in \text{Free}(\{D_1, D_6, D_8, \overleftrightarrow{P_2}\})$, then $G \in \text{Free}(\{D_1, \ldots, D_8\})$ and is G a directed co-graph.Since $G \in \text{Free}(\{\overleftrightarrow{P_2}\})$, there is no series operation in any construction of G which implies that G is an oriented co-graph. (3) $\Leftrightarrow$ (4) Since co-graphs are precisely the P_4-free graphs [CLSB81]. (2) $\Rightarrow$ (7) By Lemma 5.2.9. (7) $\Rightarrow$ (2) If G is transitive, then $G \in \text{Free}(\{D_1, D_6\})$.(1) $\Leftrightarrow$ (5) and (1) $\Leftrightarrow$ (6) By [GWY16]. (1)&(2) $\Rightarrow$ (4) By Observation 5.2.8. (3) $\Rightarrow$ (2) If $und(G)$ does not contain a P_4, then G can not contain any orientation of P_4. $\square$

Among others are two subclasses of oriented co-graphs, which will be of interest within our results. By restricting within Definition 5.2.5 (2) to $k = 2$ and graph G_1 or G_2 to an edgeless graph or to a single vertex, we obtain the class of all *oriented simple co-graphs* or *oriented threshold graphs*, respectively. The class of oriented threshold graphs has been introduced by Boeckner in [Boe18].

5.2.2 Directed width parameters and digraph operations

We will now regard how different directed width parameters behave regarding the above mentioned digraph operations and transformation. These results can then be used to extend Bodlaender and Möhrings results on undirected graphs to directed graphs and show, that several directed graph parameters are computable in linear/polynomial time on directed co-graphs and further, that some directed graph parameters are equal on directed co-graphs.

Directed path-width

In order to prove our main results, we show some properties of directed path-decompositions. Similar results are known for undirected path-decompositions and are useful within several places.

Lemma 5.2.11 ([YC08]). *Let G be some digraph and H be a subdigraph of G, then d-pw$(H) \leq$ d-pw(G).*

Lemma 5.2.12 (Bidirectional complete subdigraph). *Let $G = (V, E)$ be some digraph, $G' = (V', E')$ with $V' \subseteq V$ be a bidirectional complete subdigraph, and $(X_1, \dots, X_r)$ a directed path-decomposition of G. Then, there is some i, $1 \leq i \leq r$, such that $V' \subseteq X_i$.*

Proof. We show the claim by an induction on $|V'|$. If $|V'| = 1$ then by (dpw-1) there is some i, $1 \leq i \leq r$, such that $V' \subseteq X_i$. Next, suppose $|V'| > 1$ and $v \in V'$. By our induction hypothesis there is some i, $1 \leq i \leq r$, such that $V' - \{v\} \subseteq X_i$. By (dpw-3) there are two integers r_1 and r_2, $1 \leq r_1 \leq r_2 \leq r$, such that $v \in X_j$ for all $r_1 \leq j \leq r_2$. If $r_1 \leq i \leq r_2$ then $V' \subseteq X_i$. Further, suppose that $i < r_1$ or $r_2 < i$. If $i < r_1$, we define $j' = r_1$ and if $i > r_2$, we define $j' = r_2$. We now show that $V' \subseteq X_{j'}$. Let $w \in V' \setminus \{v\}$. Since there are two arcs (v, w) and (w, v) in E, by (dpw-2) there is some $r_1 \leq j'' \leq r_2$ such that $v, w \in X_{j''}$. By (dpw-3) we conclude $w \in X_{j'}$. Thus, $V' \setminus \{v\} \subseteq X_{j'}$ and $\{v\} \subseteq X_{j'}$, i.e. $V' \subseteq X_{j'}$. □

Lemma 5.2.13 (Bidirectional complete bipartite subdigraph). *Let $G = (V, E)$ be a digraph and $(X_1, \dots, X_r)$ a directed path-decomposition of G. Further, let $A, B \subseteq V$, $A \cap B = \emptyset$, and $\{(u, v), (v, u) \mid u \in A, v \in B\} \subseteq E$. Then, there is some i, $1 \leq i \leq r$, such that $A \subseteq X_i$ or $B \subseteq X_i$.*

Proof. Suppose that $B \not\subseteq X_i$ for all $1 \leq i \leq r$. Then, there are $b_1, b_2 \in B$ and $i_{1,\ell}, i_{1,r}, i_{2,\ell}, i_{2,r}$, $1 \leq i_{1,\ell} \leq i_{1,r} < i_{2,\ell} \leq i_{2,r} \leq r$, such that $\{i \mid b_1 \in X_i\} = \{i_{1,\ell}, \dots, i_{1,r}\}$ and $\{i \mid b_2 \in X_i\} = \{i_{2,\ell}, \dots, i_{2,r}\}$ (and both sets are disjoint). Let $a \in A$. Since $(b_2, a) \in E$ there is some $i_{2,\ell} \leq i \leq r$ such that $a \in X_i$ and since $(a, b_1) \in E$ there is some $1 \leq j \leq i_{1,r}$ such that $a \in X_j$. By (dpw-3) it is true that $a \in X_k$ for every $i_{1,r} \leq k \leq i_{2,\ell}$.

If we suppose $A \not\subseteq X_i$ for all $1 \leq i \leq r$, it follows that $b \in X_k$ for every $i_{1,r} \leq k \leq i_{2,\ell}$. □

Lemma 5.2.14. *Let $\mathcal{X} = (X_1, \dots, X_r)$ be a directed path-decomposition of some digraph $G = (V, E)$. Further, let $A, B \subseteq V$, $A \cap B = \emptyset$, and $\{(u, v), (v, u) \mid u \in A, v \in B\} \subseteq E$. If there is some i, $1 \leq i \leq r$, such that $A \subseteq X_i$ then there are $1 \leq i_1 \leq i_2 \leq r$ such that*

1. *for all i, $i_1 \leq i \leq i_2$ is $A \subseteq X_i$,*
2. *$B \subseteq \cup_{i=i_1}^{i_2} X_i$, and*

3. *$\mathcal{X}' = (X'_{i_1}, \dots, X'_{i_2})$ where $X'_i = X_i \cap (A \cup B)$ is a directed path-decomposition of the digraph induced by $A \cup B$.*

Proof. Let $i_1 = \min\{i \mid A \subseteq X_i\}$ and $i_2 = \max\{i \mid A \subseteq X_i\}$. Since $\mathcal{X}$ satisfies (dpw-3), statement (1.) holds.

Since there is some i, $1 \leq i \leq r$, such that $A \subseteq X_i$, we know that $\mathcal{X} = (X_1, \dots, X_r)$ is also a directed path-decomposition of $G' = (V, E')$ where $E' = E \cup \{(u, v) \mid u, v \in A, u \neq v\}$. For every $b \in B$ the digraph with vertex set $\{b\} \cup A$ is bidirectional complete subdigraph of G' which implies by Lemma 5.2.12 that there is some i, $i_1 \leq i \leq i_2$ such that $A \cup \{b\} \subseteq X_i$. Thus, there is some i, $i_1 \leq i \leq i_2$ such that $b \in X_i$ which leads to (2.).

In order to show (3.) we observe that for the sequence $\mathcal{X}' = (X'_{i_1}, \dots, X'_{i_2})$ condition (dpw-1) holds by (1.) and (2.).

By (1.) and (2.) the arcs between two vertices from A and the arcs between a vertex from A and a vertex from B satisfy (dpw-2). So let $(b', b'') \in E$ such that $b', b'' \in B$. By (2.) we know that $b' \in X_i$ and $b'' \in X_j$ for $i_1 \leq i, j \leq i_2$. If $j < i$ then by (dpw-3) for $\mathcal{X}$ there is some $X_{j'}$, $j' > i_2$ such that $b'' \in X_{j'}$ but by (dpw-3) for $\mathcal{X}$ is $b'' \in X_i$.

Further, $\mathcal{X}'$ satisfies (dpw-3) since $\mathcal{X}$ satisfies (dpw-3). □

Theorem 5.2.15. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. *$\text{d-pw}(\bullet) = 0$*
2. *$\text{d-pw}(G \oplus H) = \max\{\text{d-pw}(G), \text{d-pw}(H)\}$*
3. *$\text{d-pw}(G \oslash H) = \max\{\text{d-pw}(G), \text{d-pw}(H)\}$*
4. *$\text{d-pw}(G \ominus H) = \max\{\text{d-pw}(G), \text{d-pw}(H)\}$*
5. *$\text{d-pw}(G \otimes H) = \min\{\text{d-pw}(G) + |V_H|, \text{d-pw}(H) + |V_G|\}$*

Proof. 1. $\text{d-pw}(\bullet) = 0$ holds by a simple directed path-decomposition.

2. In order to show $\text{d-pw}(G \oplus H) \leq \max\{\text{d-pw}(G), \text{d-pw}(H)\}$ we consider a directed path-decomposition $(X_1, \dots, X_r)$ for G and a directed path-decomposition $(Y_1, \dots, Y_s)$ for H. Then $(X_1, \dots, X_r, Y_1, \dots, Y_s)$ leads to a directed path-decomposition of $G \oplus H$.

   Since G and H are induced subdigraphs of $G \oplus H$, by Lemma 5.2.11 the directed path-width of both digraphs leads to a lower bound on the directed path-width for the combined digraph.

3. By the same arguments as used for 2.

4. By the same arguments as used for 2.

5. In order to show $\text{d-pw}(G \otimes H) \leq \text{d-pw}(G) + |V_H|$ let $(X_1, \ldots, X_r)$ be a directed path-decomposition of G. Then, we obtain by $(X_1 \cup V_H, \ldots, X_r \cup V_H)$ a directed path-decomposition of $G \otimes H$. In the same way a directed path-decomposition of H leads to a directed path-decomposition of $G \otimes H$ which implies that $\text{d-pw}(G \otimes H) \leq \text{d-pw}(H) + |V_G|$. Thus, $\text{d-pw}(G \otimes H) \leq \min\{\text{d-pw}(G) + |V_H|, \text{d-pw}(H) + |V_G|\}$.

   For the reverse direction let $\mathcal{X} = (X_1, \ldots, X_r)$ be a directed path-decomposition of $G \otimes H$. By Lemma 5.2.13 we know that there is some i, $1 \leq i \leq r$, such that $V_G \subseteq X_i$ or $V_H \subseteq X_i$. We assume that $V_G \subseteq X_i$. We apply Lemma 5.2.14 using $G \otimes H$ as digraph, $A = V_G$ and $B = V_H$ in order to obtain a directed path-decomposition $\mathcal{X}' = (X'_{i_1}, \ldots, X'_{i_2})$ for $G \otimes H$ where for all i, $i_1 \leq i \leq i_2$, it holds that $V_G \subseteq X_i$ and $V_H \subseteq \cup_{i=i_1}^{i_2} X_i$. Further, $\mathcal{X}'' = (X''_{i_1}, \ldots, X''_{i_2})$, where $X''_i = X'_i \cap V_H$ is a directed path-decomposition of H. Thus, there is some i, $i_1 \leq i \leq i_2$, such that $|X_i \cap V_H| \geq \text{d-pw}(H) + 1$. Since $V_G \subseteq X_i$, we know that $|X_i \cap V_H| = |X_i| - |V_G|$ and $|X_i| \geq |V_G| + \text{d-pw}(H) + 1$. Thus, the width of directed path-decomposition $(X_1, \ldots, X_r)$ is at least $\text{d-pw}(H) + |V_G|$.

   If we assume that $V_H \subseteq X_i$ it follows that the width of directed path-decomposition $(X_1, \ldots, X_r)$ is at least $\text{d-pw}(G) + |V_H|$.

This shows the statements of the theorem. □

Directed tree-width

In order to show our main results, we first show some properties of directed tree-decompositions.

Lemma 5.2.16 (Bidirectional complete subdigraph). *Let $(T, \mathcal{X}, \mathcal{W})$, $T = (V_T, E_T)$, where r_T is the root of T, be a directed tree-decomposition of some digraph $G = (V, E)$ and $G' = (V', E')$ with $V' \subseteq V$ be a bidirectional complete subdigraph. Then, $V' \subseteq W_{r_T}$ or there is some $(r, s) \in E_T$, such that $V' \subseteq W_s \cup X_{(r,s)}$.*

Proof. First, we choose a vertex s in V_T, such that $W_s \cap V' \neq \emptyset$ but for every vertex s' such that $s < s'$ it holds that $W_{s'} \cap V' = \emptyset$.

Next, we show that W_s leads to a set which shows the statement of the lemma. If s is the root of T, then $W_{s'} \cap V' \neq \emptyset$ for none of its successors s' in T i.e. $W_{s'} \cap V' = \emptyset$ for all of its successors s' in T, which implies by (dtw-1) that $V' \subseteq W_s$. Otherwise, let r be the predecessor of s in T. If $V' \subseteq W_s$ the statement is true. Otherwise, let $c \in V' \setminus W_s$ and $c' \in V' \cap W_s$. Then, $(c, c') \in E$ and $(c', c) \in E$ implies that $c \in X_{(r,s)}$ by (dtw-2), since otherwise (c', c, c') is a directed walk in $G - X_{(r,s)}$ with first and last vertex $c' \in W_{\geq s}$ that uses a vertex of $G - (X_{(r,s)} \cup W_{\geq s})$, namely c. □

Lemma 5.2.17 (Bidirectional complete bipartite subdigraph). *Let $G = (V, E)$ be some digraph, $(T, \mathcal{X}, \mathcal{W})$, $T = (V_T, E_T)$, where r_T is the root of T, be a directed tree-decomposition of G. Further, let $A, B \subseteq V$, $A \cap B = \emptyset$, and $\{(u, v), (v, u) \mid u \in A, v \in B\} \subseteq E$. Then, $A \cup B \subseteq W_{r_T}$ or there is some $(r, s) \in E_T$, such that $A \subseteq W_s \cup X_{(r,s)}$ or $B \subseteq W_s \cup X_{(r,s)}$.*

Proof. Similar as in the proof of Lemma 5.2.16 we can find a vertex s in V_T, such that $W_s \cap (A \cup B) \neq \emptyset$ but for every vertex s' with $s < s'$ holds $W_{s'} \cap (A \cup B) = \emptyset$.

If s is the root of T, then $W_{s'} \cap (A \cup B) \neq \emptyset$ for none of its successors s' in T, i.e. $W_{s'} \cap (A \cup B) = \emptyset$ for all of its successors s' in T, which implies by (dtw-1) that $A \cup B \subseteq W_s$.

Otherwise, let r be the predecessor of s in T. If $A \cup B \subseteq W_s$ the statement is true. Otherwise, we know that either there is some $a \in A \cap W_s$ and $b \in B \setminus W_s$ or $a \in A \setminus W_s$ and $b \in B \cap W_s$.

We assume that there is some $a \in A \cap W_s$ and $b \in B \setminus W_s$. Then, $(a, b) \in E$ and $(b, a) \in E$ implies that $b \in X_{(r,s)}$ by (dtw-2). Thus, we have shown $B \subseteq W_s \cup X_{(r,s)}$.

If we assume that there some $b \in B$ such that $b \in W_s$, we conclude $A \subseteq W_s \cup X_{(r,s)}$. □

Lemma 5.2.18. *Let $G = (V, E)$ be a digraph of directed tree-width at most k, such that there is a 2-partition (V_1, V_2) of V with $V_1 \neq \emptyset$, $V_2 \neq \emptyset$ and $\{(u, v), (v, u) \mid u \in V_1, v \in V_2\} \subseteq E$. Let $(T, \mathcal{X}, \mathcal{W})$, $T = (V_T, E_T)$ be a directed tree-decomposition of width k for G with $|W| \leq 1$ for all $W \in \mathcal{W}$. Then, it holds that either*

(i) $\forall t \in V_T$ *with* $W_t \subseteq V_1$: $|W_t \cup \bigcup_{e \sim t} X_e \cup V_2| \leq k$

(ii) $\forall t \in V_T$ *with* $W_t \subseteq V_2$: $|W_t \cup \bigcup_{e \sim t} X_e \cup V_1| \leq k$

To prove this Lemma we first need some claims. Therefore, let $G = (V, E)$ be a digraph as in the statement of the Lemma.

Claim 5.2.19. If $(T, \mathcal{X}, \mathcal{W})$ has width $|V| - 1$, for all $t \in V_T$ it holds that $|W_t \cup \bigcup_{e \sim t} X_e \cup V_2 \cup V_1| \leq |V| - 1 = k$.

By this claim, the Lemma holds true for $k = |V| - 1$. So in all further claims we assume that the width k of $(T, \mathcal{X}, \mathcal{W})$ is smaller than $|V| - 1$.

We further assume w.l.o.g. that for all leafs ℓ of T, $W_\ell \neq \emptyset$.

Claim 5.2.20. For $k < |V| - 1$ every vertex $s \in V_T$ with $W_{>s} \cap V_1 \neq \emptyset$ and $W_{>s} \cap V_2 \neq \emptyset$ has exactly one successor t such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$. It further holds that $W_{>s} \setminus W_{\geq t} \subseteq V_1$ or $W_{>s} \setminus W_{\geq t} \subseteq V_2$.

Proof. We show this Claim in two steps.

- We first show that s has at most one successor t, such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$. Assume that s has two successors t_1 and t_2 such that $W_{\geq t_1} \cap V_1 \neq \emptyset$ and $W_{\geq t_1} \cap V_2 \neq \emptyset$ and $W_{\geq t_2} \cap V_1 \neq \emptyset$ and $W_{\geq t_2} \cap V_2 \neq \emptyset$. Then, it holds that $V \setminus W_{\geq t_1} \subseteq X_{(s,t_1)}$ and $V \setminus W_{\geq t_2} \subseteq X_{(s,t_2)}$. As $W_{\geq t_1} \cap W_{\geq t_2} = \emptyset$ it follows that $V \setminus W_{\geq t_1} \cup V \setminus W_{\geq t_2} = V$ and thus $W_s \cup \bigcup_{e \sim s} X_e = V$. Then the resulting width of $(T = (V_T, E_T), \mathcal{X}, \mathcal{W})$ is $|V| - 1$ which is a contradiction to the assumption that the width $k < |V| - 1$.
- We now show that s has at least one successor t, such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$. Assume that for all successors t of s it holds that either $W_{\geq t} \subseteq V_1$

or $W_{\geq t} \subseteq V_2$. As $W_{>s} \cap V_1 \neq \emptyset$ and $W_{>s} \cap V_2 \neq \emptyset$ there exist successors t_1, t_2 of s such that $W_{\geq t_1} \subseteq V_1$ and $W_{\geq t_2} \subseteq V_2$. Then, it holds that $V_2 \subseteq X_{(s,t_1)}$ and $V_1 \subseteq X_{(s,t_2)}$ and thus, that $V \subseteq W_s \cup \bigcup_{e \sim s} X_e$. Then, the resulting width of $(T = (V_T, E_T), \mathcal{X}, \mathcal{W})$ is $|V| - 1$ which is a contradiction to the assumption that the width $k < |V| - 1$.

As we have now proven that there is exactly one successor t such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$, for all other successors t_i of s it holds that $W_{\geq t_i} \subseteq V_1$ or $W_{\geq t_i} \subseteq V_2$. But by the same argumentation as in the second bullet point, if there are successors t_i and t_j of s such that $W_{\geq t_i} \subseteq V_1$ and $\bigcup_{\tilde{t} \geq t_j} W_{\tilde{t}} \subseteq V_2$, it would follow that $k = |V| - 1$. As this is a contradiction we can conclude that $W_{>s} \backslash W_{\geq t} \subseteq V_1$ or $W_{>s} \backslash W_{\geq t} \subseteq V_2$. □

Claim 5.2.21. Assume that $k < |V| - 1$. Let $\mathcal{L} \subseteq V_T$ be the set of all leaves ℓ of T and $L = \bigcup_{\ell \in \mathcal{L}} W_\ell$, such that for the directed path $(u_1, \ldots, u_q)$ starting at root u_1 and ending with leaf $\ell = u_q$ it holds that $\bigcup_{1 \leq i \leq q} W_{u_i} \cap V_1 \neq \emptyset$ and $\bigcup_{1 \leq i \leq q} W_{u_i} \cap V_2 \neq \emptyset$ and $\forall u_i, 1 \leq i \leq q$:

- $W_{\geq u_i} \backslash W_{\geq u_{i+1}} \subseteq V_1$ or
- $W_{\geq u_i} \backslash W_{\geq u_{i+1}} \subseteq V_2$.

Then $\mathcal{L} \neq \emptyset$ and it holds that either $L \subseteq V_1$ or $L \subseteq V_2$.

Proof. We search set $\mathcal{L}$ by traversing T starting at the root and choosing all possible paths to leafs that fulfill the conditions above.
Let u_1 be the root of T. Obviously, it holds that $W_{\geq u_1} \cap V_1 \neq \emptyset$ and $W_{\geq u_1} \cap V_2 \neq \emptyset$. For all u_i with only one successor we choose this successor u_{i+1}.
By Claim 5.2.20 for every u_i with more than one successor and $W_{\geq u_i} \cap V_1 \neq \emptyset$ and $W_{\geq u_i} \cap V_2 \neq \emptyset$ there is exactly one successor t such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$. In this case we take this t as u_{i+1}.
For u_i with $W_{\geq u_i} \cap V_1 \neq \emptyset$ or $W_{\geq u_i} \cap V_2 \neq \emptyset$ it holds that $W_{\geq u_i} \subseteq V_1$ or $W_{\geq u_i} \subseteq V_2$. Then, we choose all paths from this u_i to any following leaf and add this leaf to $\mathcal{L}$. Then, it holds that $L \subseteq V_1$ or $L \subseteq V_2$ and by construction for every $\ell \in \mathcal{L}$ it holds that for the path $(u_1, \ldots, u_q)$ from the root to ℓ it holds that $\forall u_i, 1 \leq i \leq q$:

- $W_{\geq u_i} \backslash W_{\geq u_{i+1}} \subseteq V_1$ or
- $W_{\geq u_i} \backslash W_{\geq u_{i+1}} \subseteq V_2$,

as we always choose the path containing vertices of V_1 and V_2, until only vertices from one of the sets are left. By Claim 5.2.20 this path is unique. Further, $\mathcal{L} \neq \emptyset$ holds since by construction at least one path, as described above, must exist. And as we can assume that there are no leafs with empty bags in T, it follows that $L \neq \emptyset$. □

In Claim 5.2.20 and Claim 5.2.21 we restricted the structure of the decomposition tree. But this does not suffice to prove the Lemma, we need further restrictions. As we cannot simply exclude this structures as we could in Claims 5.2.20 and 5.2.21, we give a way to transform the decomposition, such that the new decomposition tree fulfils more structural characteristics.

Claim 5.2.22. Let $k < |V| - 1$. We can assume that for $(T, \mathcal{X}, \mathcal{W})$ it holds that if $L \subseteq V_1$ ($L \subseteq V_2$ respectively), then all $W_s \subseteq V_1$ ($W_s \subseteq V_2$ respectively) with $W_{>s} \cap V_1 \neq \emptyset$ and $W_{>s} \cap V_2 \neq \emptyset$ have exactly one successor t such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$ and it further holds that $W_{>s} \setminus W_{\geq t} \subseteq V_1$ (V_2 respectively).

Proof. We show that, if $(T, \mathcal{X}, \mathcal{W})$ does not fulfill this claim, we can transform it to a decomposition $(T', \mathcal{X}', \mathcal{W}')$ of width k' such that $k' \leq k$ and all former claims remain true.

For every vertex $v \in V_T$ we define $w(v) = W_v \cup \bigcup_{e \sim_T v} X_e$ (the set that determines the width of this tree-decomposition) and for $v \in V_{T'}$ we define $w'(v)$ respectively.

Without loss of generality we assume that $L \subseteq V_1$. (As the proof for $L \subseteq V_2$ works analogously.

By Claim 5.2.20 we know that s has exactly one successor t such that $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$. We further know that $W_{>s} \setminus W_{\geq t} \subseteq V_1$ or $W_{>s} \setminus W_{\geq t} \subseteq V_2$.

The only thing which now remains open to show this Claim is that for $(T', \mathcal{X}', \mathcal{W}')$ it holds that if $W'_s \subseteq V_1$ it holds that $\bigcup_{\tilde{s} > s} W'_{\tilde{s}} \setminus \bigcup_{\tilde{t} \geq t} W'_{\tilde{t}} \subseteq V_1$.

To prove this we assume the contrary and then transform the decomposition such that the Claim holds true. So therefore we assume that in T it does not hold that $\bigcup_{\tilde{s} > s} W'_{\tilde{s}} \setminus \bigcup_{\tilde{t} \geq t} W'_{\tilde{t}} \subseteq V_1$, which by Claim 5.2.20 means, that $W_{>s} \setminus W_{\geq t} \subseteq V_2$. Let p be the predecessor of s in T and $t, t_1, \ldots, t_r$ the successors of s in T. Then, we construct $T' = (V'_T, E'_T)$ with $V'_T = V_T$ and $E'_T = \{(u, v) \mid (u, v) \in E_T, u, v \neq s\} \cup \{(t_1, s), (s, t), (p, t_1)\} \cup \bigcup_{2 \leq i \leq r} \{(t_1, t_i)\}$. In words this means means that t_1 is now a successor of p in T' and the predecessor of s and $t_2, \ldots, t_r$ in T'. Further, it holds that $W'_v = W_v$ for all $v \in V'_T$ and that $X'_e = X_e$ for all $e \in E'_T \cap E_T \setminus \{(t_1, v) \mid (t_1, v) \in E_T\}$. Let $X'_{(p,t_1)} = X_{(p,s)}$, $X'_{(t_1,s)} = X_{(s,t)} \setminus W_s$, $X'_{(t_1,t_i)} = X_{(s,t_i)}$ and for all successors v of t_1 in T let $X'_{(t_1,v)} = X_{(t_1,v)}$.

We briefly show that all conditions of a directed tree-decomposition remain fulfilled. As the W-sets does not change, it is obvious that they form a partition of V. Remains to show, that for all edges $(u, v) \in V'_T$ the set $W'_{\geq v}$ remains $X'_{(u,v)}$-normal.

- For all arcs $(u, v) \in V'_T$ with $(u, v) \in V_T$ and $W'_{\geq v} = W_{\geq v}$ and $X'_{(u,v)} = X_{(u,v)}$ it holds that $W'_{\geq v}$ is $X'_{(u,v)}$-normal, as $W_{\geq v}$ is $X_{(u,v)}$-normal
- (p, t_1): $W'_{\geq t_1} = W_{\geq s}$ is $X_{(p,s)} = X_{(p,t_1)}$-normal
- (t_1, s): $W'_{\geq s} = W_{\geq t} \cup W_s$. It holds that $W_{\geq t}$ is $X_{(s,t)}$-normal and thus $W_{\geq t} \cup W_s$ is $X_{(s,t)} \cup W_s$-normal. It follows that $W'_{\geq s} = W_{\geq t} \cup W_s$ is $X'_{(t_1,s)} = X_{(s,t)} \cup W_s$-normal.

We show that for every vertex v in V'_T it holds that $|w'(v')| - 1 \leq k$ by showing that for every vertex $v' \in V'_T$ there exists a vertex $u \in V_T$ such that $|w'(v)| \leq |w(u)|$. As for all other vertices it holds that $w'(v) = w(v)$, we only need to look at the widths induced by p, s and t.

- Consider $w'(p)$. As $W'_{>p} = W_{>p}$, it is possible to set $X'_{(p,t_1)} = X_{(p,s)}$. Then, it holds that $w'(p) = w(p)$ and further that $w'(p) - 1 \leq k$.

- Consider $w'(t_1) = W'_{t_1} \cup X'_{(p,t_1)} \cup X'_{(t_1,s)} \cup \bigcup_{v \in N_T^+(t_1)} X'_{(t_1,v)} \cup \bigcup_{2 \leq i \leq r} X'_{(t_1,t_i)}$.

  As $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$, it holds that $X_{(s,t)} = V \setminus W_{\geq t}$, so $W_{t_1} \subseteq X_{(s,t)}$. It further holds that $X_{(t_1,\tilde{t_1})} \setminus W_{t_1} \subseteq X_{s,t_1}$, such that $X_{(t_1,\tilde{t_1})} \subseteq X_{s,t_1} \cup W_{t_1}$ for all successors $\tilde{t_1}$ of t_1 in T. Thus, it follows that

$$\begin{array}{lllllll} w'(t_1) \\ = W'_{t_1} & \cup & X'_{(p,t_1)} & \cup & X'_{(t_1,s)} & \cup & \bigcup_{v \in N_T^+(t_1)} X'_{(t_1,v)} & \cup & \bigcup_{2 \leq i \leq r} X'_{(t_1,t_i)} \\ = W_{t_1} & \cup & X_{(p,s)} & \cup & \left(X_{(s,t)} \setminus W_s\right) & \cup & \bigcup_{v \in N_T^+(t_1)} X_{(t_1,v)} & \cup & \bigcup_{2 \leq i \leq r} X_{(s,t_i)} \\ \subseteq X_{(s,t)} & \cup & X_{(p,s)} & \cup & X_{(s,t)} & \cup & X_{(s,t_1)} \cup W_{t_1} & \cup & \bigcup_{2 \leq i \leq r} X_{(s,t_i)} \\ \subseteq X_{(s,t)} & \cup & X_{(p,s)} & \cup & X_{(s,t)} & \cup & X_{(s,t_1)} \cup X_{(s,t)} & \cup & \bigcup_{2 \leq i \leq r} X_{(s,t_i)} \\ \subseteq W_s \cup \bigcup_{e \sim_T s} X_e \\ = w(s) \end{array}$$

- Consider $w'(s) = W'_s \cup X'_{(t_1,s)} \cup X'_{(s,t)}$. As $W_{\geq t} \cap V_1 \neq \emptyset$ and $W_{\geq t} \cap V_2 \neq \emptyset$, it holds that $X'_{(t_1,s)} = V \setminus W_{\geq s} \subset V \setminus W_{\geq t} = X'_{(s,t)}$. As further $X'_{(s,t)} = X_{(s,t)}$ it follows that

$$w'(s) = W'_s \cup X'_{(t_1,s)} \cup X'_{(s,t)} \subseteq X'_{(s,t)} = X_{(s,t)} \subseteq w(s).$$

Thus, the widths induced by the vertices v with $w'(v) \neq w(v)$, are not increasing the width of the directed tree-decomposition. □

Proof. of Lemma 5.2.18. By Claim 5.2.19 the Lemma holds for $k = |V| - 1$. Assume that $k < |V| - 1$. Let L be the set of all leaves ℓ in T such that for the path $(u_1, \ldots u_q)$ from the root to ℓ it holds that $\bigcup_{1 \leq i \leq q} W_{u_i} \cap V_1 \neq \emptyset$, $\bigcup_{1 \leq i \leq q} W_{u_i} \cap V_2 \neq \emptyset$ and $\forall u_i$ with $1 \leq i \leq q$:

- $W_{\geq u_i} \setminus W_{\geq u_{i+1}} \subseteq V_1$ or
- $W_{\geq u_i} \setminus W_{\geq u_{i+1}} \subseteq V_2$.

By Claim 5.2.21 it holds that $L \neq \emptyset$ and either $L \subseteq V_1$ or $L \subseteq V_2$.
W.l.o.g. assume that $L \subseteq V_1$, for $L \subseteq V_2$ the proof works analogously.
We show that $\forall W_t \in \mathcal{W}$ with $W_t \subseteq V_1$ it holds that $W_t \cup \bigcup_{e \sim t} X_e \cup V_2 \leq k$. By the construction in Claim 5.2.21 and the structural information of Claim 5.2.22, there is a vertex u_r such that $L \subseteq W_{>u_r}$, $W_{>u_r} \subseteq V_1$ and $W_{u_r} \subseteq V_2$. Let $(u_1, \ldots u_r)$ be the path from the root to this vertex. As $W_{>u_r} \subseteq V_1$, it holds that for all successors u of u_r that $V_2 \subseteq X_{(u_r,u)}$ and as $W_{u_r} \subseteq V_2$, it holds that $V \setminus W_{\geq u_r} \subseteq X_{(u_{r-1},u_r)}$. It then holds that $V_2 \cup (V \setminus W_{\geq u_r}) \subseteq W_{u_r} \cup \bigcup_{e \sim u_r} X_e$ and thus $|V_2 \cup (V \setminus W_{\geq u_{r-1}})| \leq k$. Further, for the path $(u_1, \ldots, u_{r-1})$ it holds that for every $1 \leq i \leq r-1$, $X_{(u_i,u_{i+1})} = V \setminus W_{\geq u_{i+1}} \subseteq V \setminus W_{\geq u_{r-1}} = X_{(u_{r-1},u_r)}$. Let now W_t be any element of $\mathcal{W}$ such that $W_t \subseteq V_1$. By Claim 5.2.21 and Claim 5.2.22 it holds that either

(i) There is a successor t' of t such that $W_{\geq t'} \subseteq V_1$ or

(ii) $t = u_i$ with $1 \leq i \leq r-2$.

In case (i) it holds that $V_2 \subseteq X_{(t,t')}$ and thus, $W_t \cup \bigcup_{e\sim t} X_e \cup V_2 = W_t \cup \bigcup_{e\sim t} X_e \leq k$. In case (ii) it holds that $X_{(u_{i-1},u_i)} = V \setminus W_{\geq u_i}$ and $X_{(u_i,u_{i+1})} = V \setminus W_{\geq u_{i+1}}$. By Claim 5.2.22 it holds that $W_{>u_i} \setminus W_{\geq u_{i+1}} \subseteq V_1$. Thus, we can assume that u_i has no other successors but u_{i+1}, as otherwise we are in case (i). If u_{i+1} is the only successor of u_i, then $W_{u_i} \cup \bigcup_{e\sim u_i} X_e = V \backslash W_{\geq u_{i+1}} \subseteq V \backslash W_{\geq u_{r-1}}$ and then $V_2 \cup W_{u_i} \cup \bigcup_{e\sim u_i} X_e \subseteq V \backslash W_{\geq u_{r-1}} \cup V_2 \subseteq W_{u_{r-1}} \cup \bigcup_{e\sim u_{r-1}} X_e$ and further $|V_2 \cup W_{u_i} \cup \bigcup_{e\sim u_i} X_e| \leq k$ holds. □

Theorem 5.2.23. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs then the following properties hold.*

1. *d-tw($\bullet$) $= 0$*
2. *d-tw($G \oplus H$) $= \max\{$d-tw(G), d-tw(H)$\}$*
3. *d-tw($G \oslash H$) $= \max\{$d-tw(G), d-tw(H)$\}$*
4. *d-tw($G \ominus H$) $= \max\{$d-tw(G), d-tw(H)$\}$*
5. *d-tw($G \otimes H$) $= \min\{$d-tw(G) $+ |V_H|$, d-tw(H) $+ |V_G|\}$*

Proof. Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs. Further, let $(T_G, \mathcal{X}_G, \mathcal{W}_G)$ be a directed tree-decomposition of G such that r_G is the root of $T_G = (V_{T_G}, E_{T_G})$ and $(T_H, \mathcal{X}_H, \mathcal{W}_H)$ be a directed tree-decomposition of H, such that r_H is the root of $T_H = (V_{T_H}, E_{T_H})$.

1. d-tw($\bullet$) $= 0$ holds by a simple directed tree-decomposition.
2. We define a directed tree-decomposition $(T_J, \mathcal{X}_J, \mathcal{W}_J)$ for $J = G \oplus H$. Let ℓ_G be a leaf of T_G. Let T_J be the disjoint union of T_G and T_H with an additional arc (ℓ_G, r_H). Further, let $\mathcal{X}_J = \mathcal{X}_G \cup \mathcal{X}_H \cup \{X_{(\ell_G,r_H)}\}$, where $X_{(\ell_G,r_H)} = \emptyset$ and $\mathcal{W}_J = \mathcal{W}_G \cup \mathcal{W}_H$. The triple $(T_J, \mathcal{X}_J, \mathcal{W}_J)$ satisfies (dtw-1) since the combined decompositions satisfy (dtw-1). Further, $(T_J, \mathcal{X}_J, \mathcal{W}_J)$ satisfies (dtw-2) since additionally in J there is no arc from a vertex of H to a vertex of G. This shows that d-tw($G \oplus H$) $\leq \max\{$d-tw(G), d-tw(H)$\}$. Since G and H are induced subdigraphs of $G \oplus H$, by Lemma 3.3.8 the directed tree-width of both leads to a lower bound on the directed tree-width for the combined digraph.
3. The same arguments lead to d-tw($G \oslash H$) $= \max\{$d-tw(G), d-tw(H)$\}$.
4. The same arguments lead to d-tw($G \ominus H$) $= \max\{$d-tw(G), d-tw(H)$\}$.
5. In order to show d-tw($G \otimes H$) $\leq$ d-tw(G) $+ |V_H|$ let T_J be the disjoint union of a new root r_J and T_G with an additional arc (r_J, r_G). Further, let $\mathcal{X}_J = \mathcal{X}'_G \cup \{X_{(r_J,r_G)}\}$, where $\mathcal{X}'_G = \{X_e \cup V_H \mid e \in E_{T_G}\}$ and $X_{(r_J,r_G)} = V_H$ and $\mathcal{W}_J = \mathcal{W}_G \cup \{W_{r_H}\}$, where $W_{r_J} = V_H$. Then, $(T_J, \mathcal{X}_J, \mathcal{W}_J)$ is a directed tree-decomposition of width at most d-tw(G) $+ |V_H|$ for $G \otimes H$.

In the same way a disjoint union of a new root r_J and T_H with an additional arc (r_J, r_H), $\mathcal{X}'_H = \{X_e \cup V_G \mid e \in E_{T_H}\}$, $X_{(r_J,r_H)} = V_G$, $W_{r_J} = V_G$ lead to a directed tree-decomposition of width at most d-tw$(H) + |V_G|$ for $G \otimes H$. Thus, d-tw$(G \otimes H) \leq \min\{$d-tw$(G) + |V_H|,$ d-tw$(H) + |V_G|\}$.

For the reverse direction let $(T_J, \mathcal{X}_J, \mathcal{W}_J)$, $T_J = (V_T, E_T)$, be a directed tree-decomposition of minimum width for $G \otimes H$. By Lemma 3.3.9 we can assume that $|W_t| \leq 1$ for every $t \in V_T$. Then, by Lemma 5.2.18 we can assume that $\forall t \in V_T$ with $W_t \subseteq V_G$ it holds that $|W_t \cup \bigcup_{e \sim t} X_e \cup V_H| \leq$ d-tw$(G \otimes H)$ or $\forall t \in V_T$ with $W_t \subseteq V_H$ it holds that $|W_t \cup \bigcup_{e \sim t} X_e \cup V_G| \leq$ d-tw$(G \otimes H)$.

We assume that $\forall t \in V_T$ with $W_t \subseteq V_H$: $|W_t \cup \bigcup_{e \sim t} X_e \cup V_G| \leq$ d-tw$(G \otimes H)$.

We define $(T'_J, \mathcal{X}'_J, \mathcal{W}'_J)$ as follows. We initialize $T'_J = (V'_T, E'_T)$ with $V'_T = V_T$, $E'_T = E_T$, $X'_e = X_e \cap V_H$, and $W'_s = W_s \cap V_H$. Whenever this leads to an empty set W'_s such that $\bigcup_{\tilde{s} \geq s} W'_{\tilde{s}} = \emptyset$, delete vertex s. Then, $(T'_J, \mathcal{X}'_J, \mathcal{W}'_J)$ is a directed tree-decomposition of H.

The width of $(T'_J, \mathcal{X}'_J, \mathcal{W}'_J)$ is at most d-tw$(G \otimes H) - |V_G|$ as shown in the following.

- Suppose s is a vertex in T'_J such that $W'_s = W_s$. Then, it holds that $W_s \subseteq V_H$ and thus $|W_s \cup \bigcup_{e \sim s} X_e \cup V_G| \leq$ d-tw$(G \otimes H)$. So it holds that $|W'_s \cup \bigcup_{e \sim s} X'_e| \leq$ d-tw$(G \otimes H) - |V_G|$.
- Suppose s is a vertex in T'_J such that $W'_s \neq W_s$. Then $W_s \subseteq V_G$ and $W_s \neq \emptyset$. By construction of T'_J, we know that $\bigcup_{\tilde{s} \geq s} W_{\tilde{s}} \cap V_H \neq \emptyset$. Now, we distinguish two cases:
  - Suppose there is a successor t of s such that $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}} \subseteq V_H$. Then $V_G \subseteq X_{(s,t)}$ and thus $|W'_s \cup \bigcup_{e \sim s} X'_e| \leq$ d-tw$(G \otimes H) - |V_G|$.
  - Suppose that for all successors t of s it holds that $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}} \not\subseteq V_H$. Then, as $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}} \cap V_H \neq \emptyset$ and $\bigcup_{\tilde{t} \geq t} W_{\tilde{t}} \cap V_G \neq \emptyset$ it follows by the same argumentation as in the proof of Lemma 5.2.18 that $|W_s \cup \bigcup_{e \sim s} X_e \cup V_G| \leq$ d-tw$(G \otimes H)$. So it holds that $|W'_s \cup \bigcup_{e \sim s} X'_e| \leq$ d-tw$(G \otimes H) - |V_G|$.

Thus, the width of $(T'_J, \mathcal{X}'_J, \mathcal{W}'_J)$ is at most d-tw$(G \otimes H) - |V_G|$ and since $(T'_J, \mathcal{X}'_J, \mathcal{W}'_J)$ is a directed tree-decomposition of H, it follows that d-tw$(H) \leq$ d-tw$(G \otimes H) - |V_G|$.

If we assume that $\forall t \in V_T$ with $W_t \subseteq V_G$: $|W_t \cup \bigcup_{e \sim t} X_e \cup V_H| \leq$ d-tw$(G \otimes H)$ or $\forall t \in V_T$, it follows that d-tw$(G) \leq$ d-tw$(G \otimes H) - |V_H|$.

This shows the statements of the theorem. □

The proof of Theorem 5.2.23 is constructive as we give a tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of minimum width for every directed co-graph. Since for any operation we define a decomposition where T is a path, we conclude that for any directed co-graph there is a tree-decomposition $(T, \mathcal{X}, \mathcal{W})$ of minimum width such that T is a path.

Next, we give some examples where the equality of directed path-width and directed tree-width does not hold, as also claimed in Proposition 3.5.25.

Example 5.2.24. Every complete biorientation of a rooted tree has directed tree-width 1 and a directed path-width depending on its height. The path-width of perfect 2-ary trees of height h is $\lceil h/2 \rceil$ (cf. [Sch89]) and for $k \geq 3$ the path-width of perfect k-ary trees of height h is exactly h by [EST94, Corollary 3.1].

Remark 5.2.25. The results of Theorem 5.2.15 and Theorem 5.2.23 imply that for every directed co-graph its directed path-width equals its directed tree-width using the definition allowing empty sets W_r of [JRST01a] (see Section 5.2.3 for details). Since d-tw$(G) \leq$ d-pw(G) for all graphs G holds for both variants of directed tree-width allowing and forbidding empty sets W_r and directed tree-width allowing empty sets W_r is smaller or equal to directed tree-width forbidding empty sets W_r, the statements of Theorem 5.2.23 also hold true when considering the directed tree-width forbidding empty sets W_r in [JRST01b].

Directed feedback vertex set number

Theorem 5.2.26. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. $\mathit{fvs}(\bullet) = 0$
2. $\mathit{fvs}(G \oplus H) = \mathit{fvs}(G) + \mathit{fvs}(H)$
3. $\mathit{fvs}(G \oslash H) = \mathit{fvs}(G) + \mathit{fvs}(H)$
4. $\mathit{fvs}(G \ominus H) = \mathit{fvs}(G) + \mathit{fvs}(H)$
5. $\mathit{fvs}(G \otimes H) = \min\{\mathit{fvs}(G) + |V_H|, \mathit{fvs}(H) + |V_G|\}$

Proof. 1. $\text{fvs}(\bullet) = 0$ holds since a single vertex digraph is a DAG.

2. Since there is no edge between a vertex from G and a vertex from H, all cycles in $G \oplus H$ are between two vertices from G or two vertices from H.

3. Since the edges from G to H do not create any new cycle in $G \oslash H$ all cycles in $G \oslash H$ are between two vertices from G or two vertices from H.

4. Same arguments as (3.)

5. In order to show $\text{fvs}(G \otimes H) \leq \min\{\text{fvs}(G) + |V_H|, \text{fvs}(H) + |V_G|\}$, we can obtain a DAG from $G \otimes H$ either by removing all vertices from H and a minimum subset $S \subset V_G$ or by removing all vertices from G and a minimum subset $S \subset V_H$.

For the reverse direction we observe the following. Since in $G \otimes H$ every vertex of V_G has an edge to and from every vertex of V_H, every subset S such that $(G \otimes H) \setminus S$ is a DAG must fulfill $V_G \subseteq S$ or $V_H \subseteq S$. If $V_G \subseteq S$, we have to remove all cycles from H which optimally can be done by $\text{fvs}(H)$ vertices and if $V_H \subseteq S$, we have to remove all cycles from G which optimally can be done by $\text{fvs}(G)$ vertices. Thus, $\min\{\text{fvs}(G) + |V_H|, \text{fvs}(H) + |V_G|\}$ leads a lower bound on $\text{fvs}(G \otimes H)$.

This shows the statements of the theorem. □

Directed feedback arc set number

Theorem 5.2.27. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. $fas(\bullet) = 0$
2. $fas(G \oplus H) = fas(G) + fas(H)$
3. $fas(G \oslash H) = fas(G) + fas(H)$
4. $fas(G \ominus H) = fas(G) + fas(H)$
5. $fas(G \otimes H) = fas(G) + fas(H) + |V_G| \cdot |V_H|$

Proof. 1. $\text{fas}(\bullet) = 0$ holds since a single vertex digraph is a DAG.

2. Since there is no edge between a vertex from G and a vertex from H, all cycles in $G \oplus H$ are between two vertices from G or two vertices from H.
3. Since the edges from G to H do not create any new cycle in $G \oslash H$ all cycles in $G \oslash H$ are between two vertices from G or two vertices from H.
4. Same arguments as (3.)
5. In order to show $\text{fas}(G \otimes H) \leq \text{fas}(G) + \text{fas}(H) + |V_G| \cdot |V_H|$, the cycles with all vertices in G can be removed by $\text{fas}(G)$ arcs, the cycles with all vertices in H can be removed by $\text{fas}(H)$ arcs, and the cycles of length 2 between a vertex of G and a vertex from H can be removed by $|V_G| \cdot |V_H|$ arcs.

   Since none of these three types of removals destroys a cycle of a further type, this number is best possible and is also a lower bound.

This shows the statements of the theorem. □

Cycle Rank

Theorem 5.2.28. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. $cr(\bullet) = 0$
2. $cr(G \oplus H) = \max\{cr(G), cr(H)\}$

3. $cr(G \oslash H) = \max\{cr(G), cr(H)\}$

4. $cr(G \ominus H) = \max\{cr(G), cr(H)\}$

5. $cr(G \otimes H) = \min\{cr(G) + |V_H|, cr(H) + |V_G|\}$

Proof. 1. $\mathrm{cr}(\bullet) = 0$ holds by definition.

2. If $G \oplus H$ is acyclic, then G and H are acyclic and thus $\mathrm{cr}(G) = \mathrm{cr}(H) = 0$ and the statement is true. Otherwise, since digraph $G \oplus H$ is not strongly connected, by definition $\mathrm{cr}(G \oplus H)$ is the maximum cycle rank of any strongly connected component of $G \oplus H$. Since every strongly connected component of $G \oplus H$ is a subset of V_G or a subset of V_H, the statement is true.

3. Holds by the same arguments as in (2.).

4. Holds by the same arguments as in (2.).

5. First, we show $\mathrm{cr}(G \otimes H) \leq \min\{\mathrm{cr}(G) + |V_H|, \mathrm{cr}(H) + |V_G|\}$. Since $G \otimes H$ is strongly connected, we can apply the second case of Definition 3.3.21 to verify an upper bound of $\mathrm{cr}(G) + |V_H|$ by removing the vertices of H one by one from $G \otimes H$ and to verify an upper bound of $\mathrm{cr}(H) + |V_G|$ by removing the vertices of G one by one from $G \otimes H$.

   For the reverse direction we observe the following. Since in $G \otimes H$, every vertex of V_G has an edge to and from every vertex of V_H such that digraph $G \otimes H$ remains strongly connected at least as long as it has vertices from V_G and V_H. Thus, we have to apply the second case of Definition 3.3.21 as long we have vertices from V_G and vertices from V_H. This either leads to a subdigraph induced by $V_G \setminus V'_G$ for some $V'_G \subset V_G$ or to a subdigraph induced by $V_H \setminus V'_H$ for some $V'_H \subset V_H$. Thus, we have

$$\begin{aligned}\mathrm{cr}(G \otimes H) &\geq \min\{|V_H| + |V'_G| + \mathrm{cr}(G - V'_G), |V_G| + |V'_H| + \mathrm{cr}(H - V'_H)\} \\ &\geq \min\{|V_H| + \mathrm{cr}(G), |V_G| + \mathrm{cr}(H)\}.\end{aligned}$$

   This shows the statements of the theorem. □

DAG-width

Lemma 5.2.29. *Let $G = (V, E)$ be a directed graph of DAG-width at most k, such that $V_1 \cup V_2 = V$, $V_1 \cap V_2 = \emptyset$, and $\{(u, v), (v, u) \mid u \in V_1, v \in V_2\} \subseteq E$. Then, there is a DAG-decomposition $(D, \mathcal{X})$, $D = (V_D, E_D)$, of width at most k for G such that for every $v \in V_D$ it holds that $V_1 \subseteq X_v$ or for every $v \in V_D$ it holds that $V_2 \subseteq X_v$.*

Proof. Let $G = (V, E)$ be a digraph of DAG-width at most k and $(D, \mathcal{X})$ be a nice DAG-decomposition of width at most k for G. Assume that $k < |V|$. Otherwise, for $k = |V|$ the DAG consisting only of one vertex v with bag $X_v = V$ would be a minimum DAG-decomposition such that for every $v \in V_D$ it holds that $V_1 \subseteq X_v$ and $V_2 \subseteq X_v$.

Since D is a DAG it allows a topological ordering. We show the claim by traversing the vertices of D in a reverse topological ordering, i.e. we start visiting the sinks of D. Let d'' be a sink of D and d' be a predecessor of d'' in D. Without loss of generality let $v \in X_{d''}$ for some $v \in V_1$. Then, by (dagw-3) it holds that $v \in X_{d'}$ or $V_2 \subseteq X_{d'} \cap X_{d''}$.

Case 1 $v \in X_{d'}$. Let d be a predecessor of d' in D. Then, by (dagw-3) it holds that $v \in X_d$ or $V_2 \subseteq X_d \cap X_{d'}$.

Case 1.1 $V_2 \subseteq X_d \cap X_{d'}$ This is equivalent to Case 2, but $X_{d'}$ is bigger or equal than in Case 2, as $\mid X_{d'} \setminus \{v\} \cup V_2 \mid \leq \mid X_{d'} \cup V_2 \mid$. So we can omit this case.

Case 1.2 $v \in X_d$ Assume we repeat this to source s because otherwise, we create an equivalence to Case 1.1. Then, for the path $(s, v_1, \ldots, d', d'')$ we have $X_{v_1} \cup \cdots \cup X_{d'} \cup X_{d''} \subseteq X_s$. Repeating this for all vertices to the source and all sinks would lead to $V_1 \cup V_2 = V \subseteq X_s$ for source s, which contradicts $k < |V|$. So for some sink ℓ this Case 1.2 cannot be repeated to the source. Regarding the omitted cases, we then know that for every predecessor p of ℓ it holds that $V_2 \in X_p$. Then, it holds that $V_2 \in X_s$. As also $X_{v_1} \cup \cdots \cup X_{d'} \cup X_{d''} \subseteq X_s$, the width of this digraph would be lower or equal by adding V_2 to $V_{v_1}, \ldots, X_{d'}, X_{d''}$. So we can omit this case.

Case 2 $V_2 \subseteq X_d \cap X_{d'}$. Let d be a predecessor of d' in D. Then, by (dagw-3) it holds that $V_2 \subseteq X_d$ or $V_1 \subseteq X_d \cap X_{d'}$.

Case 2.1 $V_1 \subseteq X_d \cap X_{d'}$ In this case, $V_1 \cup V_2 \subseteq X_{d'}$. This is a contradiction to $k < |V|$, so we can omit this case.

Case 2.2 $V_2 \subseteq X_d$ By using this argument recursively, for every predecessor p of d it holds that $V_1 \subseteq X_p$. Using this argumentation on all sinks of D, for every path $(s, v_1, \ldots, v_i)$ from source s to a sink v_i it holds that $V_1 \subseteq X_s, X_{v_1}, \ldots, X_{v_i}$ or $V_2 \subseteq X_s, X_{v_1}, \ldots, X_{v_i}$. Because of minimality of X_s, it follows that $V_2 \subseteq X_v$ for all $v \in V_D$.

This shows the statements of the lemma. □

Obviously, this lemma also holds for a nice DAG-decomposition.

Theorem 5.2.30. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. $dagw(\bullet) = 1$
2. $dagw(G \oplus H) = \max\{dagw(G), dagw(H)\}$
3. $dagw(G \oslash H) = \max\{dagw(G), dagw(H)\}$
4. $dagw(G \ominus H) = \max\{dagw(G), dagw(H)\}$
5. $dagw(G \otimes H) = \min\{dagw(G) + |V_H|, dagw(H) + |V_G|\}$

Proof. Let G and H be two vertex-disjoint digraphs and let further $(D_G, \mathcal{X}_G)$ and $(D_H, \mathcal{X}_H)$ be their nice DAG-decompositions with minimum DAG-width. Let r_H be the root of D_H and let ℓ_G be a leaf of D_G.

1. $\text{dagw}(\bullet) = 1$ holds by a simple DAG-decomposition.

2. For $J = G \oplus H$, we first define a DAG-decomposition $(D_J, \mathcal{X}_J)$ for J and show that it is of minimum width afterwards. Let D_J be the disjoint union of D_G and D_H with an additional arc (ℓ_G, r_H). Further, $\mathcal{X}_J = \mathcal{X}_G \cup \mathcal{X}_H$. $(D_J, \mathcal{X}_J)$ is a valid DAG-decomposition because it satisfies the conditions as follows. It holds that (dagw-1) is satisfied by $(D_G, \mathcal{X}_G)$ and $(D_H, \mathcal{X}_H)$ it is also satisfied by $(D_J, \mathcal{X}_J)$ because all vertices of J are included. As we do not add any vertices to the X-sets and G and H are vertex-disjoint, (dagw-2) is satisfied for $(D_J, \mathcal{X}_J)$. Further, (dagw-3) is satisfied for all arcs in D_G and D_H. In D_J there is only one additional arc, (ℓ_G, r_H). Since it holds that for r_H, $X_{\succcurlyeq r_H}$ is guarded by $\emptyset$ and we do not add any outgoing vertices to H and $X_{\ell_G} \cap X_{r_H} = \emptyset$, (dagw-3) is satisfied for $(D_J, \mathcal{X}_J)$. Thus, the DAG-width of the decomposition is limited by the larger width of G and H, such that $\text{dagw}(G \oplus H) \leq \max\{\text{dagw}(G), \text{dagw}(H)\}$.

   The lower bound holds as G and H are both induced subdigraphs of J and a digraph cannot have lower DAG-width than its induced subdigraphs. Hence $\text{dagw}(J) \geq \max\{\text{dagw}(G), \text{dagw}(H)\}$ applies, which then leads to $\text{dagw}(J) = \max\{\text{dagw}(G), \text{dagw}(H)\}$.

3. Holds by the same arguments as given in (2.).

4. Holds by the same arguments as given in (2.).

5. For $J = G \otimes H$, set $D_J = D_G$ and $\mathcal{X}_J = \{X_u \cup V_H \mid X_u \in \mathcal{X}_G\}$. Then, $(D_J, \mathcal{X}_J)$ is a DAG-decomposition for J: Obviously, (dagw-1) is satisfied. (dagw-2) and (dagw-3) are satisfied since they are satisfied for $\mathcal{X}_G$ and we add V_H to every vertex set in $\mathcal{X}_G$. Further, it holds that the width of $(D_J, \mathcal{X}_J)$ is $\text{dagw}(G) + |V_H|$. In the same way, we get a DAG-decomposition of width $\text{dagw}(H) + |V_G|$, so we have $\text{dagw}(G \otimes H) \leq \min\{\text{dagw}(G) + |V_H|, \text{dagw}(H) + |V_G|\}$.

   We use Lemma 5.2.29 for a lower bound. Assume $\text{dagw}(G \otimes H) < \min\{\text{dagw}(G) + |V_H|, \text{dagw}(H) + |V_G|\}$. Let $(D_J, \mathcal{X}_J)$ be a minimum DAG-decomposition of J of width $k < \min\{\text{dagw}(G) + |V_H|, \text{dagw}(H) + |V_G|\}$. By Lemma 5.2.29 we have $V_H \subseteq X_v$ for all $X_v \in \mathcal{X}_J$ or $V_G \subseteq X_v$ for all $X_v \in \mathcal{X}_J$. Without loss of generality assume $V_H \subseteq X_v$ for all $X_v \in \mathcal{X}_J$ (because $V_G \subseteq X_v$ for all $X_v \in \mathcal{X}_J$, respectively). Then, $(D'_G, \mathcal{X}'_G)$ with $D'_G = D_J$, $\mathcal{X}'_G = \{X_u \setminus V_H \mid X_u \in \mathcal{X}_J\}$ is a DAG-decomposition of width $k - |V_H|$ of G:

   – (dagw-1) is satisfied since $\bigcup_{u \in V_{D'_G}} X_u = \bigcup_{u \in V_{D_J}} (X_u \setminus V_H) = \left(\bigcup_{u \in V_{D_J}} X_u\right) \setminus V_H = V_J \setminus V_H = (V_G \cup V_H) \setminus V_H = V_G$.
   – (dagw-2) is satisfied since for all $u, v, w \in V_{D'_G}$ with $u \succcurlyeq_{D'_G} v \succcurlyeq_{D'_G} w$ and X^J_u, X^J_v and X^J_w the corresponding sets in $(D_J, \mathcal{X}_J)$ it holds that $X_u \cap$

$X_w = \left(X_u^J \setminus V_H\right) \cap \left(X_w^J \setminus V_H\right) = \left(X_u^J \cap X_w^J\right) \setminus V_H \subseteq X_v^J \setminus V_H = X_v$ as $u \succcurlyeq_{D_J} v \succcurlyeq_{D_J} w$.

– (dagw-3) is satisfied since for all edges $(u,v) \in E_{D'_G}$, we have $(u,v) \in E_{D_J}$ and as $X_u \cap X_v = \left(X_u^J \cap X_v^J\right) \setminus V_H$ which guards $X_{\succcurlyeq_{D'_G} v} \setminus X_u = X_{\succcurlyeq_{D_J} v} \setminus X_u^J$. For the root the condition is trivially satisfied.

But it holds that $k - |V_H| < \min\{\text{dagw}(G) + |V_H|, \text{dagw}(H) + |V_G|\} - |V_H| \leq \text{dagw}(G) + |V_H| - |V_H| = \text{dagw}(G)$. This is a contradiction, as it is not possible to create a DAG-decomposition of width smaller than dagw(G). Thus, it follows that $\text{dagw}(G \otimes H) \geq \min\{\text{dagw}(G) + |V_H|, \text{dagw}(H) + |V_G|\}$.

This shows the statements of the theorem. □

Kelly-width

Theorem 5.2.31. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. $kw(\bullet) = 1$
2. $kw(G \oplus H) = \max\{kw(G), kw(H)\}$
3. $kw(G \oslash H) = \max\{kw(G), kw(H)\}$
4. $kw(G \ominus H) = \max\{kw(G), kw(H)\}$
5. $\max\{kw(G), kw(H)\} \leq kw(G \otimes H) \leq \min\{kw(G) + |V_H|, kw(H) + |V_G|\}$

Proof. We use the fact that by Lemma 3.3.34, a digraph has Kelly-width $k+1$ if and only if it has a directed elimination ordering of width k. Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs with $\text{kw}(G) = k_G$ and $\text{kw}(H) = k_H$. Then, there exists a directed elimination ordering $\lhd_G$ of G of width $k_G - 1$ and a directed elimination ordering $\lhd_H$ of H of width $k_H - 1$.

1. $\text{kw}(\bullet) = 1$ holds by a simple Kelly decomposition.
2. For $J = G \oplus H$, we obtain a linear ordering $\lhd_J$ of J by adding first all vertices from $\lhd_H$ and from $\lhd_G$ to $\lhd_J$ afterwards. As no edges from H to G are inserted to J, this is a directed elimination ordering of width $\max\{k_H - 1, k_G - 1\}$. As G and H are both induced subdigraphs of J, there cannot exist a directed elimination ordering of smaller width. By Lemma 3.3.34 it follows that $\text{kw}(J) = \max\{k_H, k_G\}$, such that $\text{kw}(G \oplus H) = \max\{\text{kw}(G), \text{kw}(H)\}$.
3. Holds by the same arguments as in (2.).
4. Holds by the same arguments as in (2.).

5. For $J = G \otimes H$, we obtain a linear ordering $\lhd_J$ of J by adding first all vertices from $\lhd_H$ and afterwards from $\lhd_G$ to $\lhd_J$ (first $\lhd_G$, then $\lhd_H$ respectively). As there are exactly V_G (V_H) more outgoing edges for every vertex in V_H (V_G), this is a directed elimination ordering of J of width $k_H - 1 + |V_G|$ ($k_G - 1 + |V_H|$, respectively).

   The lower bound holds as G and H are both induced subdigraphs of J and a digraph cannot have lower Kelly-width than its induced subdigraphs.

This shows the statements of the theorem. □

Remark 5.2.32. The value $\min\{\mathrm{kw}(G)+|V_H|, \mathrm{kw}(H)+|V_G|\}$ is not a lower bound for $\mathrm{kw}(G \otimes H)$, even not if G and H are directed co-graphs. Figure 5.3 shows two isomorphic digraphs G and H of Kelly-width 3, by the elimination orderings (a, b, c, d, e) and (f, g, h, i, j) and the induced 4-cliques. But $G \otimes H$ has Kelly-width at most 7, by the elimination ordering $(a, f, b, c, d, e, g, h, i, j)$, so we have

$$\mathrm{kw}(G \otimes H) = 7 < 8 = \min\{\mathrm{kw}(G) + |V_H|, \mathrm{kw}(H) + |V_G|\}.$$

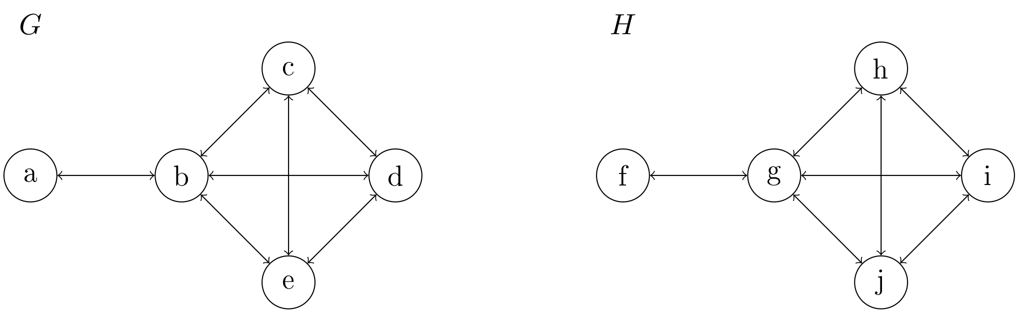


Figure 5.3: Two isomorphic digraphs G and H of Kelly-width 3.

DAG-depth

Theorem 5.2.33. *Let $G = (V_G, E_G)$ and $H = (V_H, E_H)$ be two vertex-disjoint digraphs, then the following properties hold.*

1. $ddp(\bullet) = 1$
2. $ddp(G \oplus H) = \max\{ddp(G), ddp(H)\}$
3. $ddp(G \oslash H) = ddp(G) + ddp(H)$
4. $\max\{ddp(G), ddp(H)\} \leq ddp(G \ominus H) \leq ddp(G) + ddp(H)$
5. $ddp(G \otimes H) = \min\{ddp(G) + |V_H|, ddp(H) + |V_G|\}$

Proof. 1. $\mathrm{ddp}(\bullet) = 1$ holds by definition.

2. To show the upper bound $\mathrm{ddp}(G \oplus H) \leq \max\{\mathrm{ddp}(G), \mathrm{ddp}(H)\}$, let F_G and F_H be directed elimination forests of minimum height for G and H, respectively. Since none of the vertices of V_G is reachable from a vertex of V_H and none of the vertices of V_H is reachable from a vertex of V_G the disjoint union of F_G and F_H leads to a directed elimination forest $F_{G\oplus H}$ for $G \oplus H$. Then, it holds that

$$\begin{aligned} \mathrm{ddp}(G \oplus H) \leq h(F_{G\oplus H}) + 1 &= \max\{h(F_G), h(F_H)\} + 1 \\ &= \max\{\mathrm{ddp}(G) - 1, \mathrm{ddp}(H) - 1\} + 1 \\ &= \max\{\mathrm{ddp}(G), \mathrm{ddp}(H)\}. \end{aligned}$$

   Further, the lower bound $\mathrm{ddp}(G\oplus H) \geq \max\{\mathrm{ddp}(G), \mathrm{ddp}(H)\}$ holds by [GHK$^+$14, Corollary 3.12] since G and H are both subdigraphs of $G \oplus H$.

3. To show the upper bound $\mathrm{ddp}(G \oslash H) \leq \mathrm{ddp}(G) + \mathrm{ddp}(H)$, let F_G and F_H be directed elimination forests of minimum height for G and H, respectively. Since the set of reachable fragments for $G \oslash H$ can be obtained by $R(G \oslash H) = \{f \cup V_H \mid f \in R(G)\}$ we obtain a directed elimination forest $F_{G\oslash H}$ for $G \oslash H$ as follows. Starting with a copy of F_G we replace every vertex (x, X) by $(x, X \cup V_H)$. Further, for every leaf ℓ of F_G we insert a copy of F_H and an edge between ℓ and every root in the corresponding copy of F_H. This leads to a directed elimination forest for $G \oslash H$ of height $h(F_G) + h(F_H) + 1$. This leads to

$$\mathrm{ddp}(G \oslash H) \leq h(F_{G\oslash H}) + 1 = h(F_G) + h(F_H) + 1 + 1 = \mathrm{ddp}(G) + \mathrm{ddp}(H).$$

   In order to show the lower bound $\mathrm{ddp}(G \oslash H) \geq \mathrm{ddp}(G) + \mathrm{ddp}(H)$, let $F_{G\oslash H}$ be a directed elimination forest of minimum height for $G \oslash H$. Since none of the vertices of V_G is reachable from a vertex of V_H, the vertices of V_H do not affect the number of fragments, reachable from V_G. Thus, we can assume that no vertex of V_H is a predecessor of a vertex of V_G in a tree of $F_{G\oslash H}$. Thus, we can a obtain directed elimination forest F_G for G by removing all vertices (x, X) from $F_{G\oslash H}$ where $x \in V_H$, as well as all vertices in V_H from all sets X. In the same way we can obtain a directed elimination forest F_H for H. This leads to

$$\mathrm{ddp}(G \oslash H) = h(F_{G\oslash H}) + 1 = h(F_G) + h(F_H) + 1 + 1 \geq \mathrm{ddp}(G) + \mathrm{ddp}(H).$$

4. Since $G \ominus H$ is a subdigraph of $G \oslash H$, we know by [GHK$^+$14, Corollary 3.12] that $\mathrm{ddp}(G \ominus H) \leq \mathrm{ddp}(G \oslash H)$ and thus, that the upper bound follows by (2.).

   Further, the lower bound $\mathrm{ddp}(G \ominus H) \geq \max\{\mathrm{ddp}(G), \mathrm{ddp}(H)\}$ holds since G and H are both subdigraphs of $G \ominus H$.

5. First, we show that $\mathrm{ddp}(G \otimes H) \leq \min\{\mathrm{ddp}(G) + |V_H|, \mathrm{ddp}(H) + |V_G|\}$. Since $G \otimes H$ has only one reachable fragment as long as it contains vertices from V_G and vertices from V_H, we can apply the second case of Definition 3.3.23 to verify an upper bound of $\mathrm{ddp}(G) + |V_H|$ by removing the vertices of H one by one from $G \otimes H$ and to verify an upper bound of $\mathrm{ddp}(H) + |V_G|$ by removing the vertices of G one by one from $G \otimes H$.

Next, we show that $\text{ddp}(G \otimes H) \geq \min\{\text{ddp}(G) + |V_H|, \text{ddp}(H) + |V_G|\}$. Since in $G \otimes H$ every vertex of V_G has an edge to and from every vertex of V_H, $G \otimes H$ has only one reachable fragment as long as it contains vertices from V_G and V_H. Thus, we have to apply the second case of Definition 3.3.23 as long as we have vertices from V_G and vertices from V_H. This either leads to a subdigraph induced by $V_G - V'_G$ for some $V'_G \subset V_G$ or to a subdigraph induced by $V_H - V'_H$ for some $V'_H \subset V_H$. Thus, we have

$$\begin{aligned}\text{ddp}(G \otimes H) &\geq \min\{|V_H| + |V'_G| + \text{ddp}(G - V'_G), |V_G| + |V'_H| + \text{ddp}(H - V'_H)\} \\ &\geq \min\{|V_H| + \text{ddp}(G), |V_G| + \text{ddp}(H)\}.\end{aligned}$$

This shows the statements of the theorem. □

Note that $\text{ddp}(G \ominus H)$ cannot be computed from $\text{ddp}(G)$ and $\text{ddp}(H)$ by a simple formula since the disjoint union and the order operation behave differently.

5.2.3 Digraph width measures on directed co-graphs

Next, we apply the results of Section 5.2.2 in order to show close relations between the considered parameters on directed co-graphs.

Theorem 5.2.34. *For every extended directed co-graph G, we have*

$$kw(G) - 1 \leq d\text{-}pw(G) = d\text{-}tw(G) = cr(G) = dagw(G) - 1 \leq fvs(G) \leq fas(G) \quad (5.5)$$

and

$$dagw(G) \leq ddp(G). \quad (5.6)$$

Proof. Let $G = (V, E)$ be some extended directed co-graph. The equations and inequations given in (5.5) can be all be shown in a similar way using the results of Section 5.2.2. We exemplify this by showing $\text{d-pw}(G) = \text{d-tw}(G)$ by induction on the number of vertices $|V|$. If $|V| = 1$, then $\text{d-pw}(G) = \text{d-tw}(G) = 0$. If $G = G_1 \oplus G_2$, then by Theorem 5.2.15 and Theorem 5.2.23 it follows:

$$\text{d-pw}(G) = \max\{\text{d-pw}(G_1), \text{d-pw}(G_2)\} = \max\{\text{d-tw}(G_1), \text{d-tw}(G_2)\} = \text{d-tw}(G).$$

For the other two operations $\oslash$ and $\otimes$ and for the transformation $\ominus$ similar relations hold. □

For the inequations given in (5.5) equality is not possible by the following examples.

- Let K'_n be the $2n$ vertex digraph which is obtained by a complete digraph K_n on n vertices and adding a pendant vertex for every of the n vertices of K_n, then for the complete biorientation $\overleftrightarrow{K'_n}$ it holds that $\text{kw}(\overleftrightarrow{K'_n} \otimes \overleftrightarrow{K'_n}) = 2n - 1 < 3n - 1 = \text{d-pw}(\overleftrightarrow{K'_n} \otimes \overleftrightarrow{K'_n})$.

- For transitive tournaments $\overrightarrow{T_n}$, $n \geq 2$, it holds that $\text{dagw}(\overrightarrow{T_n}) = 1 < n = \text{ddp}(\overrightarrow{T_n})$.
- For the disjoint union of two $\overleftrightarrow{K_n}$, $n \geq 3$, it holds that $\text{dagw}(2\overleftrightarrow{K_n}) = n < 2n-2 = \text{fvs}(2\overleftrightarrow{K_n})$.
- For a $\overleftrightarrow{K_n}$, $n \geq 3$, it holds that $\text{fvs}(\overleftrightarrow{K_n}) = n-1 < \frac{n(n-1)}{2} = \text{fas}(\overleftrightarrow{K_n})$.

Furthermore, the two inequations (5.5) and (5.6) cannot be combined by the following examples.

- For transitive tournaments $\overrightarrow{T_n}$, $n \geq 1$, it holds that $\text{fas}(\overrightarrow{T_n}) = 0 < n = \text{ddp}(\overrightarrow{T_n})$.
- For the disjoint union of $\ell \geq 3$ many $\overleftrightarrow{K_n}$, $n \geq 3$, it holds that $\text{ddp}(\ell\overleftrightarrow{K_n}) = n < \ell \cdot n - \ell = \text{fvs}(\ell\overleftrightarrow{K_n})$.

Theorem 5.2.35. *For every extended directed co-graph $G = (V, E)$ which is given by a binary ex-di-co-tree, the directed path-width, directed tree-width, directed feedback vertex set number, directed feedback arc set number, cycle rank, and DAG-width can be computed in time $\mathcal{O}(|V|)$.*

Proof. For all mentioned width measures the statement can be shown in a similar way using the results of section 5.2.2. We exemplify this by the result for directed path-width. Let G be some extended directed co-graph and T_G be a di-co-tree rooted at r for G. Then, algorithm Directed Path-width(r), shown in Figure 5.4, returns the directed path-width of G. The correctness follows by Theorem 5.2.15. The necessary sizes of the subdigraphs defined by subtrees of di-co-tree T_G can be precomputed in time $\mathcal{O}(|V|)$. □

Algorithm Directed Path-width(v)

```
if v is a leaf of di-co-tree T_G
    then d-pw(G[T_v]) = 0
    else {
        Directed Path-width(v_ℓ)                  ▶ v_ℓ is the left successor of v
        Directed Path-width(v_r)                  ▶ v_r is the right successor of v
        if v corresponds to a ⊕, a ⊘, or a ⊖ operation
            then
                d-pw(G[T_v]) = max{d-pw(G[T_{v_ℓ}]), d-pw(G[T_{v_r}])}
            else                                  ▶ v corresponds to a ⊗ operation
                d-pw(G[T_v]) = min{d-pw(G[T_{v_ℓ}]) + |V_{G[T_{v_r}]}|, d-pw(G[T_{v_r}]) + |V_{G[T_{v_ℓ}]}|}
    }
```

Figure 5.4: Computing the directed path-width of G for every vertex of a di-co-tree T_G.

Since di-co-trees can be computed in linear time and Theorem 5.2.33 allows to compute the DAG-depth for directed co-graphs operations we obtain the following result.

Theorem 5.2.36. *For every directed co-graph $G = (V, E)$ the directed path-width, directed tree-width, directed feedback vertex set number, directed feedback arc set number, cycle rank, DAG-width, and DAG-depth can be computed in time $\mathcal{O}(|V| + |E|)$.*

Please note that, as we can not compute ex-di-co-trees in linear time, these results can not be expanded to extended directed co-graphs.

By Lemma 3.2.4 and Lemma 3.3.7 the equality of directed path-width and directed tree-width for directed co-graphs generalizes the known results from [BM93] but cannot be obtained by the known results.

For general digraphs d-pw(G) leads to a lower bound for pw($und(G)$) by Lemma 3.2.3 and d-tw(G) leads to a lower bound for tw($und(G)$) by Lemma 3.3.6. For directed co-graphs we obtain a closer relation as follows.

Corollary 5.2.37. *Let G be a directed co-graph and $\overleftrightarrow{\omega}(G)$ be the order of a largest bidirectional complete subdigraph of G. Then, it holds that*

$$\overleftrightarrow{\omega}(G) = \textit{d-pw}(G) - 1 = \textit{d-tw}(G) - 1 \leq \textit{pw}(\textit{und}(G)) - 1 = \textit{tw}(\textit{und}(G)) - 1 = \omega(\textit{und}(G)).$$

All values are equal if and only if G is a complete bioriented digraph.

Proof. The equality $\text{pw}(und(G)) - 1 = \text{tw}(und(G)) - 1 = \omega(und(G))$ has been shown in [BM93]. The equality $\overleftrightarrow{\omega}(G) = \text{d-pw}(G) - 1 = \text{d-tw}(G) - 1$ follows by Lemma 5.2.12 (or Lemma 5.2.16) and Theorem 5.2.34. The upper bound follows by Lemma 3.2.3 or Lemma 3.3.6. □

The relations between digraph width measures given in Table 3.5 can be improved when restricting to directed co-graphs as shown in Table 5.1.

$$\begin{array}{ccccccccccccc} & & & & & & & & \text{ddp} & & & & \\ & & & & & & & & \downarrow & & & & \\ \text{d-tw} & \longleftrightarrow & \text{dagw} & \longleftrightarrow & \text{kw} & \longleftrightarrow & \text{d-pw} & \longleftrightarrow & \text{cr} & \longleftarrow & \text{fvs} & \longleftarrow & \text{fas} \end{array}$$

Table 5.1: Known relations between digraph width measures restricted to directed co-graphs. A directed edge from measure β to measure α indicates that by the results summarized in Table 3.5 and Theorem 5.2.34 there has been shown some function f such that for every directed co-graph it holds $\alpha(G) \leq f(\beta(G))$.

5.2.4 Graph Coloring on Recursively Defined Digraphs

We now consider graph coloring problems on recursively defined digraphs, especially on oriented and directed co-graphs. Huge parts of this subsection are taken from [GKR19b] and [GKR21a].

Oriented Graph Coloring for Oriented Graphs

First, we give some results on the oriented graph coloring for general recursively defined oriented graphs. These results will be very useful to prove our results for oriented co-graphs.

Please note that this section is taken in huge parts from [GKR19b].

Lemma 5.2.38. *Let $G_1, \ldots, G_k$ be k vertex-disjoint oriented graphs. Then the following equations holds:*

1. $\chi_o(G_1 \oplus \bullet) = \chi_o(G_1)$
2. $\chi_o(G_1 \oplus \ldots \oplus G_k) \geq \max\{\chi_o(G_1), \ldots, \chi_o(G_k)\}$
3. $\chi_o(G_1 \oslash \ldots \oslash G_k) = \chi_o(G_1) + \ldots + \chi_o(G_k)$

Proof.

1. $\chi_o(G_1 \oplus \bullet) \leq \chi_o(G_1)$

   Since no new arcs are inserted G_1 can keep its colors. The added isolated vertex gets a color of G_1 in order to obtain a valid coloring for $G_1 \oplus \bullet$.

   $\chi_o(G_1 \oplus \bullet) \geq \chi_o(G_1)$

   This relation holds by Lemma 3.4.3, since G_1 is an induced subdigraph of $G_1 \oplus \bullet$.

2. $\chi_o(G_1 \oplus \ldots \oplus G_k) \geq \max\{\chi_o(G_1), \ldots, \chi_o(G_k)\}$

   Since the digraphs $G_1, \ldots, G_k$ are induced subdigraphs of digraph $G_1 \oplus \ldots \oplus G_k$, all values $\chi_o(G_1), \ldots, \chi_o(G_k)$ lead to a lower bound for the number of necessary colors of the combined graph by Lemma 3.4.3.

3. $\chi_o(G_1 \oslash \ldots \oslash G_k) \leq \chi_o(G_1) + \ldots + \chi_o(G_k)$

   For $1 \leq i \leq k$ let $G_i = (V_i, A_i)$ and $c_i : V_i \to \{1, \ldots, \chi_o(G_i)\}$ a coloring for G_i. For $G_1 \oslash \ldots \oslash G_k = (V, A)$ we define a mapping $c : V \to \{1, \ldots, \sum_{j=1}^{k} \chi_o(G_i)\}$ as follows.

   $$c(v) = \begin{cases} c_1(v) & \text{if } v \in V_{G_1} \\ c_i(v) + \sum_{j=1}^{i-1} \chi_o(G_i) & \text{if } v \in V_{G_i},\ 2 \leq i \leq k. \end{cases}$$

   The mapping c satisfies the definition of an oriented coloring, because no two adjacent vertices from G_i, $1 \leq i \leq k$, have the same color by assumption and by definition of c. For $1 \leq i \neq j \leq k$ a vertex of G_i and a vertex of G_j are always adjacent, but never colored equally by definition of c.

   Further, the arcs between two color classes of every G_i, $1 \leq i \leq k$, have the same direction by definition of c. For $1 \leq i \neq j \leq k$ the arcs between a color class of G_i and a color class of G_j have the same direction by definition of the order operation.

   $\chi_o(G_1 \oslash \ldots \oslash G_k) \geq \chi_o(G_1) + \ldots + \chi_o(G_k)$

Since every G_i, $1 \leq i \leq k$, is an induced subdigraph of the combined graph, all values $\chi_o(G_1), \ldots, \chi_o(G_k)$ lead to a lower bound for the number of necessary colors of the combined graph by Lemma 3.4.3. Further, the order operations implies that for every $1 \leq i \neq j \leq k$ no vertex in G_i can be colored in the same way as a vertex in G_j. Thus, $\chi_o(G_1) + \ldots + \chi_o(G_k)$ leads to a lower bound for the number of necessary colors of the combined graph.

This shows the statements of the lemma. □

By Lemma 5.2.38, we can solve oriented coloring for oriented simple co-graphs and thus, also for subclasses, such as oriented threshold graphs and transitive tournaments, in linear time.

Proposition 5.2.39. *Let G be an oriented simple co-graph. Then, it holds that $\chi_o(G) = \chi(und(G)) = \omega(und(G))$ and all values can be computed in linear time.*

It is not easy to generalize these results to oriented co-graphs. To do so, we would need to compute the oriented chromatic number of the disjoint union of two oriented co-graphs with at least two vertices. But it is not possible to compute this oriented chromatic number of the disjoint union of general oriented graphs from the oriented chromatic numbers of the involved graphs. In Lemma 5.2.38 (2) we only show a lower bound. The following example proves that in general this can not be strengthened to equality.

Figure 5.5: Special oriented graphs: oriented cycle $\overrightarrow{C_3}$ and transitive tournament $\overrightarrow{T_3}$.

Example 5.2.40. The two graphs $\overrightarrow{C_3}$ and $\overrightarrow{T_3}$ in Figure 5.5 have the same oriented chromatic number $\chi_o(\overrightarrow{C_3}) = \chi_o(\overrightarrow{T_3}) = 3$, but their disjoint union needs more colors.

Oriented Graph Coloring for Oriented Co-Graphs

On the other hand, there are several examples for which the disjoint union does not need more than $\max\{\chi_o(G_1), \chi_o(G_2)\}$ colors, such as the union of two isomorphic oriented graphs. By Theorem 5.2.10, we know that $\overrightarrow{T_3}$, shown in Figure 5.5, is an oriented co-graph, but $\overrightarrow{C_3}$, shown in Figure 5.5, is not an oriented co-graph. Consequently, the question arises whether oriented coloring could be closed under disjoint union, when restricted to oriented co-graphs.

In order to solve OCN restricted to oriented co-graphs G we created an algorithm, which is shown in Figure 5.6. The method traverses a canonical di-co-tree T for G using a depth-first search, such that for every inner vertex the children are visited from left to right. For every inner vertex u of T, we store two values in$[u]$ and out$[u]$. These values ensure that the vertices of G, corresponding to the leaves of the subtree, rooted at u will we labeled by labels ℓ, such that $\text{in}[u] \leq \ell \leq \text{out}[u]$. For every leaf vertex u of T, we additionally store the label of the corresponding vertex of G in $\text{color}[u]$. These values lead to an optimal oriented coloring of G by the next theorem.

Algorithm LABEL(G, u, i)

```
if (u is a leaf of T) {
    color[u] = i; in[u] = i; out[u] = i;
}
else {
    in[u] = i; out[u] = 0;
    for all children v of u from left to right do {
        j =Label(G, v, i);
        if (out[u] < j)
            out[u] = j;
        if (u corresponds to a disjoint union)
            i = in[u];
        else        ▶ u corresponds to an order operation
            i = out[v] + 1;
    }
}
return out[u];
```

Figure 5.6: Computing an oriented coloring for an oriented co-graph.

Theorem 5.2.41. *Let G be an oriented co-graph. Then, an optimal oriented coloring for G and $\chi_o(G)$ can be computed in linear time.*

Proof. Let $G = (V, A)$ be an oriented co-graph. Using the method of [CP06] we can build a di-co-tree T with root r for G in linear time. Further by Lemma 5.2.7, we can assume that T is a canonical di-co-tree. For some node u of T we define by T_u the subtree of T which is rooted at u and by G_u the subgraph of G which is defined by T_u. Obviously, for every vertex u of T the tree T_u is a di-co-tree for the digraph G_u which is also an oriented co-graph.

Next, we verify that algorithm LABEL$(G, r, 1)$, shown in Figure 5.6, returns the value $\chi_o(G)$ and computes an oriented coloring for G within array $\text{color}[u]$. Therefore, we recursively show for every vertex u of T that after performing LABEL(G, u, i) for all leaves u of T_u the value $\text{color}[u]$ leads to an oriented coloring of G_u using the colors $\{i = \text{in}[u], \ldots, \text{out}[u]\}$[1] and the value $\text{out}[u] - \text{in}[u] + 1$ leads to the oriented chromatic number of G_u.

We distinguish the following three cases depending on the type of operation corresponding to the vertices u of T.

- If u is a leaf of T, then $\text{color}[u] = \text{out}[u] = \text{in}[u]$ by the algorithm leads to an oriented coloring of G_u.

[1]Please note that using colors starting at values greater than 1 is not a contradiction to Definition 3.4.1.

Further, $\text{out}[u] - \text{in}[u] + 1 = 1$, which obviously corresponds to the oriented chromatic number of G_u.

- Let u be an inner vertex of T which corresponds to an order operation and $u_1, \ldots, u_\ell$ are the children of u in T.

  We already know that the oriented colorings of G_{u_i}, $1 \leq i \leq \ell$, are feasible. Further, for $1 \leq i \neq j \leq \ell$, the algorithm's way of working ensures that a vertex from G_{u_i} and a vertex from G_{u_j} are never colored equally in G_u. For $1 \leq i \neq j \leq \ell$, the arcs between a color class of G_{u_i} and a color class of G_{u_j} have the same direction by the definition of the order operation.

  By the algorithm, value $\text{out}[u] - \text{in}[u] + 1$ is equal to $\sum_{i=1}^{\ell} \chi_o(G_{u_i})$. By Lemma 5.2.38, we conclude that $\text{out}[u] - \text{in}[u] + 1$ is equal to $\chi_o(G_{i_1} \oslash \ldots \oslash G_{i_\ell}) = \chi_o(G_u)$.

- Let u be an inner vertex of T which corresponds to a disjoint union operation and $u_1, \ldots, u_\ell$ are the children of u in T.

  We already know that the oriented colorings of G_{u_i}, $1 \leq i \leq \ell$, are feasible. Since a disjoint union operation does not create any arcs, no two adjacent vertices have the same color in G_u. Further, our method ensures that for every arc (u, v) in G it holds that $\text{color}[u] < \text{color}[v]$. Thus, all arcs between two color classes in G_u have the same direction.

  By the algorithm, value $\text{out}[u] - \text{in}[u] + 1$ is equal to $\max\{\chi_o(G_1), \ldots, \chi_o(G_\ell)\}$. By Lemma 5.2.38, we conclude that $\text{out}[u] - \text{in}[u] + 1 \leq \chi_o(G_1 \oplus \ldots \oplus G_\ell) = \chi_o(G_u)$. The relation $\text{out}[u] - \text{in}[u] + 1 \geq \chi_o(G_1 \oplus \ldots \oplus G_\ell) = \chi_o(G_u)$ holds by the feasibility of our oriented coloring.

By applying the invariant for $u = r$, the statements of the theorem follow. □

Please note that this result could be generalized in [GKL20] regarding transitive acyclic graphs.

Example 5.2.42. We illustrate the method given in Figure 5.6 by computing an oriented coloring for the oriented co-graph G, which is given by the canonical di-co-tree T of Figure 5.7. On the left of each vertex u of T, the values $\text{in}[u]$ and $\text{out}[u]$ are given. An optimal oriented coloring for G is given in blue letters below the leaves of T. The root r of T leads to $\chi_o(G) = \text{out}[r] = 5$.

Next, we can improve the result of Lemma 5.2.38(2) for oriented co-graphs.

Corollary 5.2.43. *Let $G_1, \ldots, G_k$ be k vertex-disjoint oriented co-graphs. Then, it holds that*

$$\chi_o(G_1 \oplus \ldots \oplus G_k) = \max\{\chi_o(G_1), \ldots, \chi_o(G_k)\}.$$

Proof. Let $G = G_1 \oplus \ldots \oplus G_k$ be an oriented co-graph and T be a di-co-tree with root r for G. The method given in Figure 5.6 computes an oriented coloring using $\chi_o(G) = \chi_o(G_1 \oplus \ldots \oplus G_k)$ colors. Further, the proof of Theorem 5.2.41 shows that $\chi_o(G_1 \oplus \ldots \oplus G_k) = \max\{\chi_o(G_1), \ldots, \chi_o(G_k)\}$. □

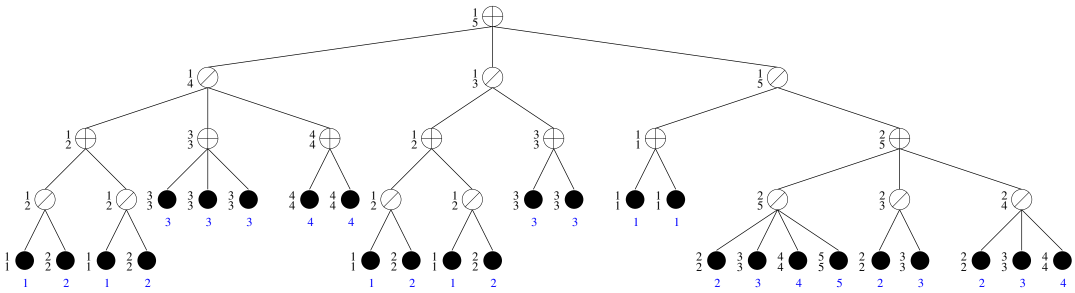


Figure 5.7: Canonical di-co-tree T for oriented co-graph G.

Corollary 5.2.44. *Let G be an oriented co-graph. The following properties are equivalent:*

1. *G is an oriented clique.*
2. *G has a di-co-tree, which does not use any disjoint union operation.*
3. *G is a transitive tournament.*

Further characterizations for transitive tournaments and oriented co-graphs, which are oriented cliques, can be found in [Gou12, Chapter 9].

As mentioned in Observation 3.4.2, oriented coloring of an oriented graph G can be characterized by the existence of homomorphisms to tournaments. These tournaments are not necessarily transitive and G is not necessarily homomorphically equivalent to some tournament. For oriented co-graphs we can show a deeper result.

Corollary 5.2.45. *There is an oriented k-coloring of an oriented co-graph G if and only if there is a homomorphism from G to some transitive tournament $\overrightarrow{T_k}$ on k vertices. Further, the oriented chromatic number of an oriented co-graph G is the minimum number k, such that G is homomorphically equivalent with the transitive tournament $\overrightarrow{T_k}$.*

Proof. Within an oriented co-graph $G = (V, A)$ the color classes $V_1, \ldots, V_k$ of an oriented k-coloring define a transitive tournament $\overrightarrow{T_k} = (\{V_1, \ldots, V_k\}, \{(V_i, V_j) \mid v_i \in V_i, v_j \in V_j, (v_i, v_j) \in A\})$. If $k = \chi_o(G)$, then there is a homomorphism from $\overrightarrow{T_k}$ to G. □

Acyclic coloring of directed co-graphs

Lemma 5.2.38 can be used to obtain the following result.

Theorem 5.2.46. *Let G be a directed co-graph. Then, an optimal acyclic coloring for G and $\vec{\chi}(G)$ can be computed in linear time.*

The *clique number* $\omega_d(G)$ of a digraph G is the number of vertices in a largest complete bioriented subdigraph of G and the *clique number* $\omega(G)$ of a (-n undirected) graph G is the number of vertices in a largest complete subgraph of G. Since the results of Lemma 5.2.38 also hold for ω_d instead of $\vec{\chi}$ we obtain the following result.

Proposition 5.2.47. *Let G be a directed co-graph. Then, it holds that*

$$\vec{\chi}(G) = \chi(und(sym(G))) = \omega(und(sym(G))) = \omega_d(G)$$

and all values can be computed in linear time.

5.2.5 Conclusion and Outlook

By that, we have shown linear time algorithms for the directed path-width, directed tree-width, directed feedback vertex set number, directed feedback arc set number, cycle rank and DAG-width of extended directed co-graphs and a linear-time algorithm for the DAG-depth of directed co-graphs. Further, we provided a comparison of all considered parameters for extended directed co-graphs and obtained equality for directed path-width, directed tree-width, cycle rank and DAG-width. Furthermore, we showed for bounds for the class of directed co-graphs for the directed vertex set number, DAG-depth and Kelly-width.

The results on directed path-width and directed tree-width generalize the equivalence of path-width and tree-width of co-graphs which is known from [BM93] to directed graphs. The shown equality can be generalized to an equivalence of directed path-width and other definitions of directed tree-width, see Theorem 3.5.23.

Our results on the width measures for the directed union of digraphs can be used to show that most of the considered width measures can be obtained by the width of of its strong components. In order to process the strong components of a digraph G, we use its *acyclic condensation* such that every digraph G can be represented by the directed union of its strong components, see section 5.1. Our results on the directed union imply that the directed path-width[2], directed tree-width, cycle rank, DAG-width, and Kelly-width of a digraph is the maximum width of its strong components. Further, the directed feedback vertex set number and the directed feedback arc set number of a digraph is the sum of the widths of its strong components.

Furthermore, we obtain that for directed co-graphs Kelly-width can be bounded by DAG-width (Theorem 5.2.34). Due to [HK08, Conjecture 30], [AKK$^+$15], and [BJG18, Section 9.2.5] this remains open for general digraphs and is related to one of the biggest open problems in graph searching, namely whether the monotonicity costs for Kelly- and DAG-width games are bounded.

It remains open whether there is a linear or polynomial time algorithm to compute Kelly-width on directed co-graphs. Furthermore, it would be interesting to know

[2] This result is known from [YC08] using the directed vertex separation number, which is equal to the directed path-width. Our results allow to show this connection directly using directed path-decompositions.

for which superclasses of directed co-graphs it is still possible to find polynomial algorithms to get the considered parameters and for which superclasses these problems become NP-hard. While the class of directed co-graphs was studied well in [CP06], for the class of extended directed co-graphs it remains to show how to compute an ex-di-co-tree in order to apply Theorem 5.2.35.

5.3 Directed Threshold Graphs

Directed threshold graphs are a subclass of directed co-graphs. They are useful to characterize digraphs of directed linear NLC-width 1 and digraphs of directed neighbourhood-width 1.

Huge parts of this section are taken from [GR19a].

Definition 5.3.1 (Directed threshold graphs)**.** The class of directed threshold graphs is recursively defined as follows.

(i) Every digraph on a single vertex $(\{v\}, \emptyset)$, denoted by $\bullet$, is a directed threshold graph.

(ii) If G is a directed threshold graph, then (a) $G \oplus \bullet$, (b) $G \oslash \bullet$, (c) $\bullet \oslash G$, and (d) $G \otimes \bullet$ are directed threshold graphs.

In Theorem 5.3.4 we will show that directed threshold graphs can be characterized by the eighteen forbidden induced subdigraphs shown in Figures 5.2 and 5.8.

The related class oriented threshold graphs was considered by Boeckner in [Boe18] by using all given operations except the series composition $G \otimes \bullet$.

Observation 5.3.2. Every oriented threshold graph is a directed threshold graph and every directed threshold graph is a directed co-graph.

Corollary 3.5.20 allows us to bound the directed path-width of directed threshold graphs as follows.

Corollary 5.3.3. *The directed path-width of a directed threshold graph G is at most* $\min(\Delta^-(G), \Delta^+(G))$.

Proof. The set of directed threshold graphs has directed linear NLC-width 1 (see Theorem 5.3.4. Thus the result follows by Corollary 3.5.20. □

Since $\Delta^-(G) \leq \Delta(G)$ and $\Delta^+(G) \leq \Delta(G)$ and thus

$$\min(\Delta^-(G), \Delta^+(G)) \leq \Delta(G)$$

the given bounds also hold for the more common measure $\Delta(G)$ instead of $\min(\Delta^-(G), \Delta^+(G))$.

Theorem 5.3.4. *For every digraph G the following statements are equivalent.*

(a) d-lnlcw$(G) = 1$.

(b) $d\text{-}nw(G) = 1$.

(c) $d\text{-}lcw(G) \leq 2$ *and* $G \in Free(\{D_2, D_4, D_9, D_{10}, D_{12}, D_{13}, D_{14}\})$.

(d) G *is a directed threshold graph.*

(e) $G \in Free(\{D_1, \ldots, D_{15}, 2\overrightarrow{P_2}, \overrightarrow{P_2} \cup \overleftrightarrow{P_2}, 2\overleftrightarrow{P_2}\})$.

(f) $G \in Free(\{D_1, \ldots, D_6, D_{10}, D_{11}, D_{13}, D_{14}, D_{15}\})$ *and* $und(G) \in Free(\{P_4, 2K_2, C_4\})$.

(g) $G \in Free(\{D_1, \ldots, D_6, D_{10}, D_{11}, D_{13}, D_{14}, D_{15}\})$ *and* $und(G)$ *is a threshold graph.*

Proof. (a) $\Leftrightarrow$ (b) By the proof of Lemma 3.5.8 (which can be proved similarly to the case of the undirected versions in [Gur06b]) the set of all digraphs of directed linear NLC-width 1 is equal to the set of all digraphs of directed neighbourhood-width 1.

(e) $\Rightarrow$ (d) If digraph G does not contain $D_1, \ldots, D_8$ (see Table 5.2), then digraph G is a directed co-graph by [CP06] and thus has a construction using disjoint union, series composition, and order composition. By excluding D_9, D_{10}, and D_{11} we know that for every series composition of G_1 and G_2 either G_1 or G_2 is bidirectional complete. Thus this subdigraph can also be added by a number of series operations with one vertex.

Further by excluding D_{12}, D_{13}, D_{14}, and D_{15} we know that for every order composition of G_1 and G_2 either G_1 or G_2 is a tournament and since we exclude a directed cycle of length 3 by D_6, we know that G_1 or G_2 even is a transitive tournament. Thus this subdigraph can also be added by a number of order operations with one vertex.

By excluding $2\overrightarrow{P_2}, \overrightarrow{P_2} \cup \overleftrightarrow{P_2}, 2\overleftrightarrow{P_2}$ for every disjoint union of G_1 and G_2 either G_1 or G_2 has no edge. Thus this subdigraph can also be added by a number of disjoint union operations with one vertex.

(a) $\Rightarrow$ (d) Let $G = (V, E)$ be a digraph of directed linear NLC-width 1 and X be a directed linear NLC-width 1-expression for G. An expression $c(X)$ using directed threshold graph operations for G can recursively be defined as follows.

- Let $X = \bullet_1$ for $t \in [k]$. Then $c(X) = \bullet$.
- Let $X = \circ_R(X')$ for $R : [1] \to [1]$. Then $c(X) = c(X')$.
- Let $X = X' \otimes_{(\overrightarrow{S}, \overleftarrow{S})} \bullet_1$ for $\overrightarrow{S}, \overleftarrow{S} \subseteq [1]^2$.
  - If $\overrightarrow{S} = \emptyset$ and $\overleftarrow{S} = \emptyset$, then $c(X)$ is the disjoint union of $c(X')$ and $\bullet$.
  - If $\overrightarrow{S} = \{(1, 1)\}$ and $\overleftarrow{S} = \emptyset$, then $c(X)$ is the order composition of $c(X')$ and $\bullet$.
  - If $\overrightarrow{S} = \emptyset$ and $\overleftarrow{S} = \{(1, 1)\}$, then $c(X)$ is the order composition of $\bullet$ and $c(X')$.
  - If $\overrightarrow{S} = \{(1, 1)\}$ and $\overleftarrow{S} = \{(1, 1)\}$, then $c(X)$ is the series composition of $c(X')$ and $\bullet$.

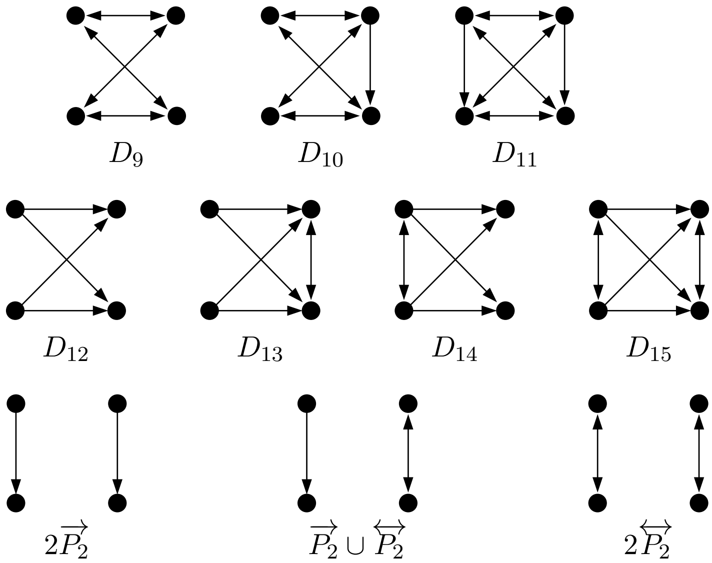


Figure 5.8: Forbidden induced subdigraphs for directed threshold graphs.

(d) $\Rightarrow$ (a) Let $G = (V, E)$ be a directed threshold graph and X be an expression using directed threshold graph operations for G. A directed linear NLC-width 1-expression $c(X)$ for G can recursively be defined as follows.

- If X defines a single vertex, then $c(X) = \bullet_1$.
- If X defines the disjoint union of expression X_1 and $\bullet$, then $c(X) = c(X_1) \otimes_{(\emptyset,\emptyset)} \bullet_1$
- If X defines the order composition of expression X_1 and $\bullet$, then $c(X) = c(X_1) \otimes_{(\{(1,1)\},\emptyset)} \bullet_1$
- If X defines the order composition of expression of $\bullet$ and X_1, then $c(X) = c(X_1) \otimes_{(\emptyset,\{(1,1)\})} \bullet_1$
- If X defines the series composition of expression X_1 and $\bullet$, then $c(X) = c(X_1) \otimes_{(\{(1,1)\},\{(1,1)\})} \bullet_1$

(d) $\Rightarrow$ (c) Digraphs $D_2, D_4, D_9, D_{10}, D_{12}, D_{13}, D_{14}$are not directed threshold graphs. Since directed threshold graphs are exactly graphs of directed linear NLC-width 1 ((a) $\Leftrightarrow$ (d)) has been shown above) by Lemma 3.5.7 we know that directed threshold graphs have directed linear clique-width at most 2.

(c) $\Rightarrow$ (e) Digraphs $D_1, D_3, D_5, D_6, D_7, D_8$have directed clique-width greater than two and thus directed linear clique-width greater than two. D_{11}, D_{15} have directed linear clique-width at least 3. Further $2\overrightarrow{P_2}, \overrightarrow{P_2} \cup \overleftrightarrow{P_2}, 2\overleftrightarrow{P_2}$ have an underlying $2K_2$ which has linear clique-width at least 3 and thus by Theorem 3.5.1(e) the directed linear clique-width of the three digraphs is also at least 3.

$(d) \Rightarrow (g)$ If G is a directed threshold graph, then $und(G)$ is a threshold graph by the recursive definition. Further the given forbidden digraphs are no directed threshold graphs and the set of directed threshold graphs is closed under taking induced subdigraphs.

$(f) \Rightarrow (e)$ For digraphs G which are excluded within (e) but not in (f), we have $und(G) \in \{P_4, C_4, 2K_2\}$.

$(f) \Leftrightarrow (g)$ Threshold graphs are exactly the set Free($\{P_4, 2K_2, C_4\}$), see [CH77]. □

Corollary 3.5.21 allows us to bound the directed path-width of planar directed threshold graphs as follows.

Corollary 5.3.5. *Planar directed threshold graphs have a directed path-width of at most* 4.

Proof. The set of directed threshold graphs has directed linear NLC-width 1 (see Theorem 5.3.4 and for planar digraphs G we know that $und(G)$ has no $K_{3,3}$ subgraph. Thus the result follows by Corollary 3.5.21. □

5.4 Twin-Distance-Hereditary Digraphs

Distance-hereditary graphs have been introduced by Howorka in 1977 [How77]. They are exactly the graphs which are distance-hereditary for their connected induced subgraphs, which means that if any two vertices u and v belong to a connected induced subgraph H of a graph G, then some shortest path between u and v in G has to be a subgraph of H.

But this is not the only definition of distance-hereditary graphs. Most important from an algorithmic perspective are the definition by forbidden induced subgraphs and the recursive construction by twins and pendant vertices. That is, a distance-hereditary graph can be defined recursively from a single vertex by the following three operations:

1. Adding a pendant vertex, which is a vertex with only one edge to an existent vertex,
2. adding a false twin, which is a vertex with the same neighborhood as an existent vertex and no edge to this vertex and
3. adding a true twin, which is a vertex with the same neighborhood as an existent vertex and an edge to this vertex.

Attempting to define a directed version of distance-hereditary graphs, it is necessary to decide which of these definitions modified to a directed definition is most promising to give an useful digraph class. We consider this matter concerning some directed graph parameters. In [LS10], the authors use a straightforward way in generalizing the property of distance-hereditary, i.e. the property that some shortest path has to be induced subgraph, on digraphs. But their graphs are limited to oriented digraphs,which are digraphs without bidirectional edges. For undirected distance-hereditary graphs, tree-width is computable in linear time [BDK00]. Further, linear rank-width of distance-hereditary graphs is computable in polynomial time [AKK17]. The clique-width of any distance-hereditary graph is at most 3 [GR99], but path-width is hard even on bipartite distance-hereditary graphs [KBMK93].

In this section we introduce a directed version of distance-hereditary graphs, which differs from the already known distance-hereditary digraphs from [LS10]. We preserve the distance-hereditary property for our new class of directed twin-distance-hereditary graphs (twin-dh digraphs for short) but we expand it as we allow bidirectional edges. These twin-dh digraphs are generated by a directed pruning sequence, as in the undirected class, by using twins and pendant vertices. This structure is algoritmically useful for showing that twin-dh digraphs have bounded clique-width. After the definition by a directed pruning sequence we go on with an other characterization of this class. Distance-hereditary graphs can be characterized by forbidden induced subgraphs, so we provide as well such a characterization for twin-dh digraphs.

Please note that huge parts of this section are taken from [KR21].

We show how to place the class in the hierarchy of related common directed graph classes and conclude that the class of twin-dh digraphs is a subclass of extended directed co-graphs, which allows to deduce some properties. Further, we take a closer look to show the connection between directed co-graphs and twin-dh digraphs and compare the class to the previously defined distance-hereditary digraphs from [LS10].

Moreover, we investigate directed width parameters on this graph class. By showing that every strong component of a twin-dh digraphs is a directed co-graph, we can prove that there are linear time algorithms to compute directed path-width, directed tree-width, DAG-width, and cycle rank on twin-dh digraphs. This further reproves the equality of these parameters, which is already given by the fact that twin-dh digraphs are also extended directed co-graphs. It generalizes our results on directed co-graphs in [GKR21b].

Furthermore, we present some properties which demonstrate the usability of the class. Showing that twin-dh digraphs have directed clique-width at most 3, it follows that for every digraph problem expressible in monadic second order logic with quantification over vertices and vertex sets there exists an fpt-algorithm with respect to the parameter directed clique-width. Thus, we can get polynomial time solutions for several different problems. From the bounded directed clique-width we can also follow that we can solve problems like Directed Hamiltonian Path, Directed Hamiltonian Cycle, Directed Cut, and Regular Subdigraph in polynomial time, following from [GWY16]. From our results in [GKR21a] we conclude that we also can solve the dichromatic number problem in polynomial time on this class.

5.4.1 Directed Distance-Hereditary Graphs

We now come to define a directed version distance-hereditary graphs. A straightforward idea given by the name of the graph class is, to say that a digraph G is called distance-hereditary, if for every induced subdigraph H of G and for every vertices u, v in H, the shortest path between u and v in H has the same length as the shortest path between u and v in G. This idea has been pursued in [LS10] but only for oriented digraphs without bioriented edges [Sch21].

In the following, we generalize the recursive definition by twins and pendant vertices to digraphs, which admits several algorithmic results.

There are at least three different definitions of twins in digraphs. In [KR09], twins have been defined to obtain distance-hereditary digraphs in context of directed rank-width and split decomposition. Thus, [KR09] can be seen as an attempt to extend undirected distance-hereditary graphs to directed distance-hereditary graphs. In [FHP19], twins have been defined to obtain results about domination and location-domination, and in [GY02] (see also [BG18, p. 282]) in studying diameter in digraphs. In [LS10] twins are introduced in context of a distance based directed version of distance-hereditary graphs, but they do not lead to a characterization of this graph class.

We define directed twins and pendant vertices in digraphs as follows.

Definition 5.4.1. Let $G = (V, E)$ be a directed graph.

- Vertices $x, y \in V$ are *directed twins*[3] if $N^-(x) \setminus \{y\} = N^-(y) \setminus \{x\}$ and $N^+(x) \setminus \{y\} = N^+(y) \setminus \{x\}$. We distinguish between
  - x is a (directed) *false twin* ($\diamond$) of y, if $(x, y), (y, x) \notin V$.
  - x is a *true out-twin* ($\leftarrow$) of y if $(y, x) \in V$, $(x, y) \notin V$.
  - x is a *true in-twin* ($\rightarrow$) of y if $(x, y) \in V$, $(y, x) \notin V$.
  - x is a *bioriented true twin* ($\leftrightarrow$) of y if $(x, y), (y, x) \in V$.
- A vertex $v \in V$ is called *pendant* if $|N^+(v)| + |N^-(v)| = 1$. We distinguish between
  - v is a *pendant plus* vertex $(+)$ if $|N^+(v)| = 1$ and $|N^-(v)| = 0$.
  - v *pendant minus* vertex $(-)$ if $|N^+(v)| = 0$ and $|N^-(v)| = 1$.

This leads to the definition of a recursively defined graph class which is close to the definition for undirected distance-hereditary graphs. We denote this class of digraphs as directed twin-distance-hereditary graphs.

Definition 5.4.2 (directed twin-distance-hereditary graphs)**.** A digraph $G = (V, E)$ is *directed twin-distance-hereditary*, twin-dh or in DDH for short, if it can be constructed recursively by taking disjoint union, adding twins and pendant vertices, starting from a single vertex.

A *directed pruning sequence* for G is a sequence $S = (s_1, \dots, s_{n-1})$, where $\sigma = (v_0, \dots, v_{n-1})$ is an ordering of V and every s_i is one of the following triples:

[3]We say twins for short, but the meaning is directed twins if the context is a digraph.

- $(v_i, +, v_{a_i})$ if v_i is a pendant plus vertex of v_{a_i} in $G[\{v_0, \ldots, v_i\}]$
- $(v_i, -, v_{a_i})$ if v_i is a pendant minus vertex of v_{a_i} in $G[\{v_0, \ldots, v_i\}]$
- $(v_i, \diamond, v_{a_i})$ if v_i is a false twin of v_{a_i} in $G[\{v_0, \ldots, v_i\}]$
- $(v_i, \leftarrow, v_{a_i})$ if v_i is a true out-twin of v_{a_i} in $G[\{v_0, \ldots, v_i\}]$
- $(v_i, \rightarrow, v_{a_i})$ if v_i is a true in-twin of v_{a_i} in $G[\{v_0, \ldots, v_i\}]$
- $(v_i, \leftrightarrow, v_{a_i})$ if v_i is a bioriented true twin of v_{a_i} in $G[\{v_0, \ldots, v_i\}]$

In general, we denote $s_i = (v_i, op_i, v_{a_i})$ and say for vertex v_i, that op_i is the *operation* and v_{a_i} the *anchor vertex* of s_i.

Like in the undirected case, for a given twin-dh digraph, it is easy to get a directed pruning sequence.

Proposition 5.4.3. *Let G be a twin-distance-hereditary digraph. Then, a directed pruning sequence of G can be computed in polynomial time.*

5.4.2 Properties of Twin-DH Digraphs

The class of directed twin-distance-hereditary graphs is closed under the connected induced subgraph operation.

Lemma 5.4.4. *Let G be a twin-dh digraph and let H be a weakly connected induced subdigraph of G. Then H is a twin-dh digraph.*

Proof. Let $G = (V, E) \in$ DDH with $V = \{v_0, \ldots, v_{n-1}\}$ and let $S(G) = (s_1, \ldots, s_{n-1})$ with $\sigma(G) = (v_0, \ldots, v_{n-1})$ be a directed pruning sequence of G. Let $H = G \setminus \{v\}$ be the weakly connected induced subdigraph H of G which emerges when deleting vertex v and all corresponding edges from G. We then create a directed pruning sequence $S(H)$ with ordering $\sigma(H)$ with the following procedures for the three different cases.

1. If $v = v_0$, we just delete s_1 from $S(G)$ to obtain $S(H)$ and adjust the indies, now v_1 is the first vertex in $\sigma(H)$.

2. If there exists $(v, op_i, a_i) \in S(G)$ and no (u_j, op_j, v) with $i < j$:
(After generating v in $S(G)$, v never occurs as an anchor vertex.)
In this case we get $S(H)$ by deleting (v, op_i, a_i) from $S(G)$ and adjust the indices.

3. If there exists $(v, op_i, a_i) \in S(G)$ and also $(u_{j1}, op_{j1}, v), ..., (u_{jk}, op_{jk}, v) \in S(G)$ with $i < j$ and $k, j \leq m - 1$:
(After generating v in $S(G)$, v occurs at least once as an anchor vertex.)
Since the emerging digraph must be weakly connected, it holds that op_{jk} must be a directed twin operation. We get a temporary $S'(H) = (s'_1, \ldots s'_{m-2})$ by the following steps.

- For $t = 1, \dots, i-1$ let $s'_t = s_t$. We keep the directed pruning sequence until v is generated.
- For $t = i$ we set $s'_i = (v', op_i, a_i)$ where v' is the vertex such that $s_h = (v', op_h, v)$ with op_h is a directed twin operation and $\nexists s_p = (v'', op_p, v)$ with $p > h$ and op_h is a directed twin operation. Thus, v' is the last twin of v with respect to $S(G)$. Now we replace v by v' as an anchor in all following occurrences. As v' is the latest twin of v w.r.t. $S(G)$, every operation applied on v is also applied on v'.
- For $t = i+1, \dots, \ell$ with $i+1 \leq \ell \leq m-1$ and $v = a_t$ first check if $v' = u_t$ in $s_t = (u_t, op_t, a_t)$. If this situation arrives, we delete this s_t from our pruning sequence, such that we set $s'_t = (,,)$. Vertex v' is now generated earlier in the directed pruning sequence and we do not need this step anymore. We will delete this empty triple at the very end, such that we don't have counting issues in the following procedure. As long as $v' \neq u_t$ we set $s'_t = s_t$ if $v \neq a_t$ and we set $s'_t = (u_t, op_t, v')$ if $v = a_t$ for $s_t = (u_t, op_t, a_t)$.
- For the remaining $t = \ell+1, \dots, m-1$ we set $s'_t = s_t$.

At the end of this procedure, we delete the empty entry $s_c = (,,)$ from $S'(H)$, adjust the indices and get a directed pruning sequence $S(H)$ for H.

This holds for every weakly connected subdigraph H, since we can repeat this procedure for every vertex which is in G but not in H. Thus, we can always get a directed pruning sequence $S(H)$ and H is a twin-dh digraph. □

As every directed pruning sequence can easily be transformed into a pruning sequence, the relation to undirected distance-hereditary graphs follows immediately.

Proposition 5.4.5. *If G is a twin-dh digraph, then $und(G)$ is distance-hereditary.*

Sub- and Superclasses of Twin-DH Digraphs

In the undirected case, distance-hereditary graphs can be classified into the hierarchy with other graph classes. Especially, they are a superclass of co-graphs by the definition of co-graphs using twins. We now show, that this is also possible in the directed case.

Proposition 5.4.6. *Every directed co-graph with at least two vertices has directed twins.*

Proof. Let G be a directed co-graph with at least two vertices. If G has exactly two vertices, then these are twins. So, let G have more than two vertices. Then $G = G_1 \star G_2$ for some directed co-graphs G_1 and G_2 with $|V(G_1)| \geq 2$ or $|V(G_2)| \geq 2$, where $\star \in \{\oplus, \oslash, \otimes\}$. By induction, G_1 or G_2 has twins x, y. Now, by definition of the $\star$-operation, x and y are also twins in G. Thus, every directed co-graph with at least two vertices has a twins as claimed.

□

Theorem 5.4.7. *A digraph is a directed co-graph if and only if it can be constructed recursively by taking disjoint union and adding directed twins, starting from a single vertex.*

Proof. Note that we may assume that all graphs considered have at least two vertices. Otherwise, the theorem clearly holds.

First, let G be a directed co-graph. Then, by Proposition 5.4.6, G has twins x and y. Let $G' = G - y$. Since G' is again a directed co-graph, by induction, G' can be constructed by taking disjoint union and adding twins, starting from single vertices. Since G is obtained from G' by adding twin y to x, G therefore can be constructed by taking disjoint union and adding twins, starting from single vertices, too.

For the other direction, suppose that G can be constructed by taking disjoint union and adding twins, starting from single vertices. We see by induction that G is a directed co-graph. Now, if G is disconnected, then, as every component of G is a directed co-graph, G is a directed co-graph. So, let us assume that G is connected. As every digraph with at most two vertices is a directed co-graph, we may also assume that G has more than two vertices. Now, by our assumption, G has twins x and y so that y is the last vertex adding to $G - y$ in obtaining G. Let $G' = G - y$. Since G' can be constructed by taking disjoint union and adding twins, G' is a directed co-graph by induction. Since G' is connected and has at least two vertices, $G' = G'_1 \star G'_2$ for some directed co-graphs G'_1 and G'_2, where $\star \in \{\oslash, \otimes\}$. Let $x \in G'_1$, say. Write $G_1 = G[V(G'_1) \cup \{y\}]$ and $G_2 = G'_2$, and note that G_1 and G_2 are directed co-graphs.

Then, since x, y are twins in G, $G = G_1 \star G_2$. Hence G is a directed co-graph, and the proof of Theorem 5.4.7 is complete.

□

Then, the relation to twin-dh digraphs follows immediately:

Corollary 5.4.8. *Let G be a directed co-graph. Then, G is also twin-distance-hereditary.*

Strong components in Twin-DH digraphs By Lemma 5.4.4 and Theorem 5.4.7, we can further conclude the following result:

Lemma 5.4.9. *Let G be a twin-dh digraph. Then every strong component of G is a directed co-graph.*

Proof. Let H be an induced subdigraph of G that is strongly connected. Then, by Lemma 5.4.4, H is a twin-dh digraph. Thus, there is a directed pruning sequence $S(H)$, that creates H. Assume that there is an element $s_i = (v_i, op_i, v_{a_i})$ in S with operation op_i is a pendant plus (respectively pendant minus) operation. Then, by the allowed operations in twin-dh digraphs, there is no directed path from v_{a_i} to v_i (respectively from v_i to v_{a_i}) in H. This is a contradiction to the fact, that H is strongly connected. Thus, S does not contain any pendant vertex operations. By Theorem 5.4.7 follows, that H is a directed co-graph. □

This lemma admits many algorithmic results. Every digraph problem, which is solvable by considering only the strong components and which is further computable on directed co-graphs, is similarly computable on twin-dh digraphs by Lemma 5.4.9. For example, this holds for several directed graph parameters, as we see later on.

With these results it also possible to show that twin-dh digraphs are a subclass of extended directed co-graphs:

Proposition 5.4.10. *Let G be a twin-dh digraph. Then G is also an extended directed co-graph.*

Proof. Let G be a twin-dh digraph. With the following procedure we can get a construction of G with the extended directed co-graph operations. We know from Lemma 5.4.9 that the strong components are directed co-graphs, thus we build the di-co-tree of these components. If a vertex does not belong to any bigger strong component it can be seen as its own strong component. The missing arc which connect the different strong components in G are built by directed union operations, where we can leave our all arcs except for the arc of the corresponding pendant vertex. □

This result allows us to deduce some results how to solve several graph parameters on this graph class. However, we show that we can even do better on twin-dh digraphs. Furthermore, twin-dh digraphs have a decisive advantage compared to its superclass since it has bounded directed clique-width. As we can build directed grids with the directed union operation in extended directed co-graphs, the directed clique-width for this class is not bounded. This allows us to solve many problems on twin-dh-digraphs, which cannot be solved on extended directed co-graphs. The reason for this is, that we lose information about the edges and therefore about the reachability within extended directed co-graphs which we preserve in the subclass.

Next, we show how this class is related to the class of distance-hereditary digraphs from [LS10].

Twin-DH Digraphs are distance-hereditary Though for the definition we used the approach of a recursive construction by twins and pendant vertices, twin-dh digraphs still fulfill the distance-heredity property.

Theorem 5.4.11. *Every twin-distance-hereditary digraph G is distance-hereditary, i.e. for every two vertices u and v in $V(G)$, all induced u,v-paths have the same length.*

In [LS10] the authors claim that for pendant vertices, for (slightly different, but more general defined) oriented twins and for false twins the distance-hereditary property remains fulfilled. However, the result that every path between two distinct vertices is of length one does not hold in general when including bioriented edges. This is why we need the following lemma, which leads us directly to the theorem above.

Lemma 5.4.12. *For two twins u, v in a twin-dh digraph G it holds that if there exists a path from u to v then the length of the shortest path in every induced subdigraph G' of G is ≤ 2.*

Note that the proof could be shortened using Theorem 4 of [LS10].

Proof. Let $u, v \in V(G)$ be twins in G. If they are bioriented twins, the distance between them is trivially 1. If u, v are oriented twins let w.l.o.g. be $(u, v) \in E(G)$. Then the distance from u to v is also 1, but this is not the case for the other direction. So let u and v be oriented twins with $(v, u) \in E(G)$ or false twins. In order to proof the lemma by contradiction, we assume that there is a shortest path from u to v of length ≥ 3 in an induced subdigraph G' of G. Let this path be $P = (u, v_1, \ldots, v_k, v)$. Since $N^-_{G'}(v) = N^-_{G'}(u)$ and $N^+_{G'}(v) = N^+_{G'}(u)$ it holds that $(v, v_1) \in E(G')$ and $(v_k, u) \in E(G')$. Then there is a cycle $(u, v_1, \ldots, v_k, u)$ of length at least 3. If the length is 3 with there must be at least two bidirectional edges in this cycle, otherwise this cycle is not constructible by directed twins. But then, one of the bidirectional edges goes to u and since v is a twin, we could have taken this shorter path (u, v_i, v) of length 2 from the beginning, which is a contradiction to the assumption of length 3. Let's assume the shortest path is > 3. Then there is a cycle $u, v_1, v_2, \ldots, v_k, u$ with the same argumentation as before. Since $und(G)$ is distance-hereditary, there cannot be any holes, thus induced cycles of length ≥ 5. Thus, the cycles must contain edges in between. If these edges are forward edges along the cycle they would shorten the path from u to v which is a contradiction. If these edges are backward edges along the cycle, they would again build smaller induced cycles, up to a $\overrightarrow{C_3}$ which is not constructible by a directed pruning sequence. Backward edges are only possible, if the outer edges from the cycle are bioriented. But this would build an induced subdigraph H_{19} or H_{16} (Fig. 5.9), which are not constructible with a directed pruning sequence and thus are not directed twin-distance-hereditary. Thus, such a path cannot exists and the shortest path is always of length ≤ 2. □

With the same example as for extended co-graphs, the class of distance-hereditary digraphs has unbounded directed clique-width. In a grid digraph, where all edges are directed from the top to the bottom and left to right, the directed clique-width increases with the number of vertices. Here we see a certain advantage of the class of twin-dh digraphs which justifies to take a closer look.

Characterization of Twin-DH Digraphs

As already mentioned previously, our definition is not only based on the property of distance heredity as in the undirected case, or regarding directed distance-hereditary graphs. That is, not every digraph which is distance-hereditary, is also a twin-dh digraph. This can be easily shown by e.g. a bioriented path. However, it is possible to give different characterizations of the class DDH by forbidden induced subdigraphs.

We give a characterization by forbidden induced subdigraphs. Therefore, we first need to define the two-leaves-digraph.

Definition 5.4.13. A weakly connected digraph G is a two-leaves-digraph if it has at least 4 vertices and if it contains at least two bioriented leaves u, v with $N(u) \neq N(v)$ in $und(G)$, see Fig. 5.9.

Theorem 5.4.14. *A digraph G is directed Twin-distance-hereditary if and only if it contains none of the the following graphs, see Fig. 5.9 as induced subdigraph.*

- $\overrightarrow{C_3}$.
- *any biorientation of the C_n (hole) for $n \geq 5$, domino, house or gem.*
- $D_3, H_1, \ldots, H_{27}$.
- *A two-leaves-digraph.*

Proof.

- $\Rightarrow$ None of the graphs can be constructed with directed twins and directed pendant vertices and the class is hereditary, see Lemma 5.4.4.
- $\Leftarrow$ We proof this by contradiction. Let G be a graph that does not contain any of the forbidden induced subgraphs above and let's assume that $G \notin$ DDH. A graph is not Twin-distance-hereditary if it cannot be constructed by the directed twin and pendant vertices operations. We distinguish two cases $G \notin$ DDH $\wedge$ $und(G) \notin$ DH and $G \notin$ DDH $\wedge$ $und(G) \in$ DH, where DH is the class of undirected distance-hereditary graphs. Case one is that G is not twin-dh for structural reasons, thus $G \notin$ DDH $\wedge$ $und(G) \notin$ DH which is ensured by the exclusion of any biorientation of the C_n (hole) for $n \geq 5$, domino, house or gem, see Fig. 5.9.
  In the other case $G \notin$ DDH$\wedge$$und(G) \in$ DH the graph is not twin-dh for orientation reasons. This means that there exists a pruning sequence P for $und(G)$ but there is no directed pruning sequence for G because the arcs have a biorientation, which cannot be achieved by the directed twin and pendant vertex operations. By forbidding the two-leaves-digraphs, the pendant vertex operations are not allowed to be bioriented and thus, every undirected pendant vertex operation in P can be replaced by a directed pendant vertex operation. It is left to show that G has as well none of the digraphs of set $\mathcal{H} = \{\overrightarrow{C_3}, D_3, H_1, \ldots, H_{27}\}$ as induced subdigraph. Note that $\mathcal{H}$ contains every graph with ≤ 4 vertices that cannot be constructed by the directed twin operations, with no inclusions. If we look at every possibly biorientation of the operations in P, we get any possible directed pruning sequence of G. For every induced subdigraph H of G with ≤ 4 vertices there must exists a biorientation, such that there is a directed pruning sequence, since the set $\mathcal{H}$ is exactly the set of graphs with $\leq$ 4 that has no directed pruning sequence. There are no more forbidden induced subdigraphs H' that contains none of the previous excluded graphs as induced subgraph with more than 4 vertices for the following reason. Assume there is an induced subdigraph H' of G with ≥ 5 vertices which is minimal in the sense that it does not contain a graph from $\mathcal{H}$ as induced subdigraph and for which there is

no directed pruning sequence. Let $V(H') = \{t_1, t_2, u, v, w_1, ..., w_k\}$ with $k \geq 1$ be the vertex set of H', where t_1 and t_2 are twins in the undirected pruning sequence P' of H'. As H' is minimal, every induced subdigraph H^* of H' with 4 vertices is not forbidden. Thus, the different directed neighborhood of t_1 and t_2 must arise by adding the fifth vertex w_1. But if this vertex causes an orientation problem in $H[\{t_1, t_2, u, v, w_1\}]$ then this vertex also causes an orientation problem in $H[\{t_1, t_2, u, w_1\}]$ which would build a forbidden induced subdigraph with 4 vertices. We end up in the same problem if we chose any other two twins. Thus, there cannot be a forbidden induced subdigraph with more than 4 vertices for which there is no directed pruning sequence, if an undirected pruning sequence exists and $G \in$ DDH.

□

To get an better understanding of the construction of the forbidden induced subdigraphs $D_3, H_1, \ldots, H_{27}$ we group them as follows. In none of them we can find directed twins, but the undirected versions of them are distance hereditary.

- $D_3, H_1, \ldots, H_5$: Digraphs with 4 or less vertices with $und(G) = C_4$ which are strongly connected.
- $H_6, \ldots, H_9$: Digraphs with 4 vertices with $und(G) = C_4$ which are not strongly connected.
- $H_{10}, \ldots, H_{19}$: Digraphs with 4 vertices with $und(G) = C_4$ with an additional single diagonal edge.
- $H_{20}, \ldots, H_{26}$: Digraphs with 4 vertices with $und(G) = C_4$ with an additional bidirectional diagonal edge.
- H_{27}: Forbidden orientation of the K_4.

5.4.3 Directed Graph Parameters on Twin-DH Digraphs

In section 5.2, we presented algorithms to compute different directed width measures on (extended) directed co-graphs in linear time. Among these are directed path-width, directed tree-width, DAG-width and cycle rank (see [GKR21b] for formal definitions). Those algorithms are not extendable directly to twin-dh digraphs, but by Lemma 5.4.9, the results can be expanded to the latter.

We have stated the following Lemma in section 5.2 for directed path-width and directed tree-width. The proof is extendable straight-forward to DAG-width and cycle rank.

Lemma 5.4.15. *The directed path-width (directed tree-width, DAG-width and cycle rank respectively) of a digraph G is the maximum of the directed path-widths (directed tree-widths, DAG-widths and cycle ranks respectively) of all strong components of G.*

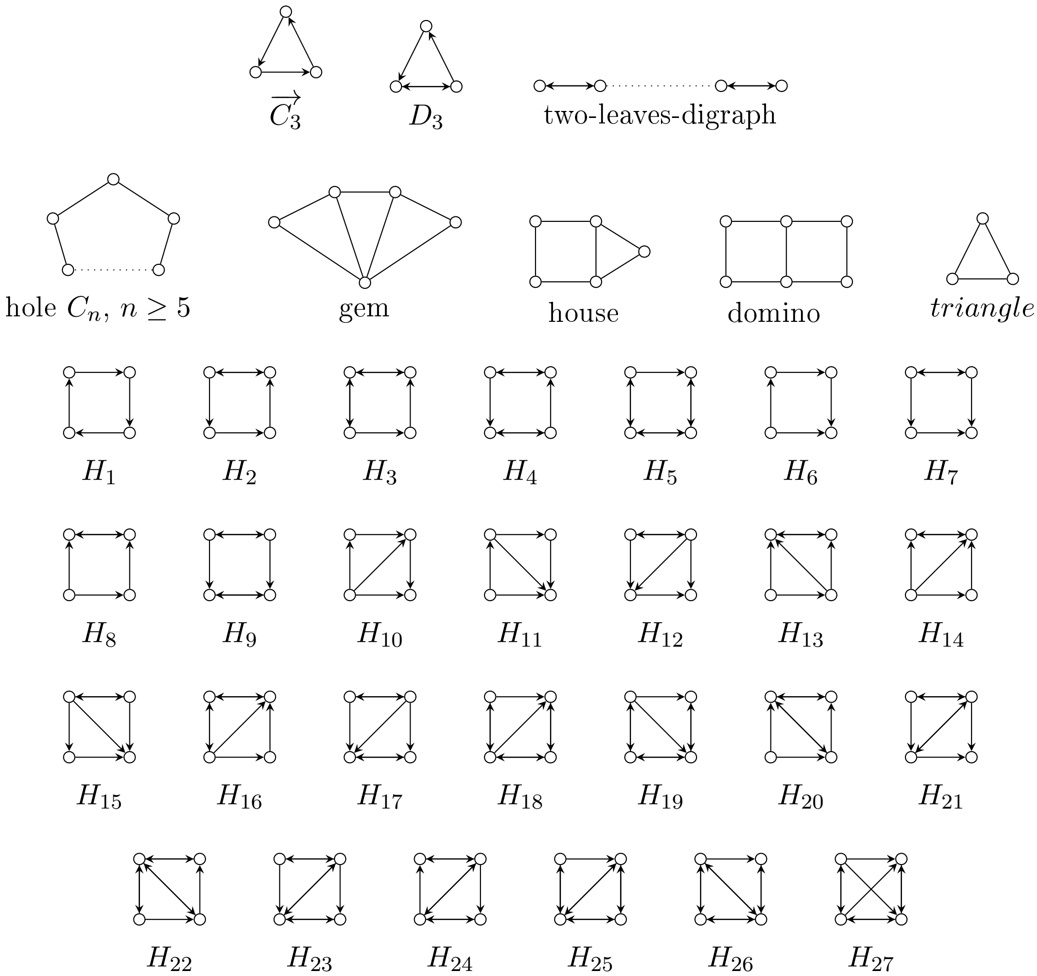


Figure 5.9: Forbidden induced sub(di)graphs.

By this lemma, we can show that it is possible to bound the computation of the mentioned parameters on a twin-distance-hereditary digraph.

Theorem 5.4.16. *Let G be a twin-distance-hereditary digraph, $n = |V(G)|$, $m = |E(G)|$. Then it holds that directed path-width (d-pw), directed tree-width (d-tw), DAG-width (dagw), and cycle rank (cr) are computable in time $\mathcal{O}(n+m)$ and further*

$$\textit{d-pw}(G) = \textit{d-tw}(G) = \textit{dagw}(G) - 1 = \textit{cr}(G). \tag{5.7}$$

Proof. It is possible to get all strong components $C_1, \ldots, C_r$ of G in linear time. By Lemma 5.4.9, all C_i, $1 \leq i \leq r$ are directed co-graphs. By section 5.2, it is possible to get the directed path-width, directed tree-width, DAG-width and cycle rank of directed co-graphs in linear time and it holds that $\text{d-pw}(C_i) = \text{d-tw}(C_i) = \text{dagw}(C_i) -$

$1 = \mathrm{cr}(C_i)$ for all $1 \le i \le r$. By Lemma 5.4.15, the directed path-width (directed tree-width, DAG-width and cycle rank respectively) of G is the maximum of the directed path-widths (directed tree-widths, DAG-widths and cycle ranks respectively) over all C_i, $1 \le i \le r$. It then follows that those parameters can be computed in linear time and that $\text{d-pw}(G) = \text{d-tw}(G) = \mathrm{dagw}(G) - 1 = \mathrm{cr}(G)$. □

Note that, as twin-dh digraphs are a subclass of extended directed co-graphs, the equality of these graph parameters follows directly from the results in [GKR21b]. However, on extended directed co-graphs (without given Ex-Di-Co-Tree), there are only known algorithms to compute them in polynomial, not linear time.

There is also a second variant to compute directed path-width on twin-dh digraphs, which generates a directed path-decomposition directly from a pruning sequence. If the pruning sequence is given, the corresponding algorithm is even faster than the method above, as its running time is only dependent from the number of vertices of the input graph, not from the edges.

Theorem 5.4.17. *Let G be a directed twin-distance-hereditary graph with given directed pruning sequence S. Let n be the number of vertices in G. Then, the directed path-width of G is computable in time $\mathcal{O}(n)$.*

We show this theorem by giving an algorithm working on the directed pruning sequence. Let therefore G be a digraph that is twin-distance-hereditary. Let $S = (s_1, \dots, s_{n-1})$ with $s_i = (v_i, op_i, v_{a_i})$ be a pruning sequence of G with ordering $\sigma = (v_0, \dots, v_{n-1})$ of the vertices in G. Note that the beginning single vertex v_0 does always exist. We further assume that for $s_1 = (v_1, op_1, v_{a_1})$ it holds that $op_1 \notin \{+, -\}$, since it is always possible to generate the second inserted vertex v_1 as a twin of v_0. Further, note that always $v_{a_1} = v_0$.

We will do a step-by-step creation to compute a directed path-decomposition from a directed pruning sequence. Therefore, we will start by the last element of the directed pruning sequence and for every element create a directed path-decomposition and then concatenate or merge it with the existing decompositions. Thus we need the following definition to concatenate and merge path-decompositions. Let

$$\mathcal{X}_u = (X_{u_1}, \dots, X_{u_a}, \dots, X_{u_b}, \dots, X_{u_r})$$

and

$$\mathcal{X}_v = (X_{v_1}, \dots, X_{v_c}, \dots, X_{v_d}, \dots, X_{v_\ell})$$

be directed path-decompositions with a function $T(\mathcal{X}_u) = \{X_{u_a}, \dots, X_{u_b}\}$ and $T(\mathcal{X}_v) = \{X_{v_c}, \dots, X_{v_d}\}$. According to this function we call $X_{u_1}, \dots, X_{u_{a-1}}, X_{v_1}, \dots X_{v_{c-1}}$ *pendant+ bags*, $X_{u_a}, \dots X_{u_b}, X_{v_c}, \dots X_{v_d}$ *twin bags* and $X_{u_{b+1}}, \dots, X_{u_r}, X_{v_{d+1}}, \dots, X_{v_\ell}$ *pendant−* bags.

Then, we define:

- The concatenation of $\mathcal{X}_u$ and $\mathcal{X}_v$ which consists of first all bags from $\mathcal{X}_u$ and then all bags from $\mathcal{X}_v$ and where the twin bags of the concatenation are exactly the twin bags of $\mathcal{X}_v$. This is denoted by:

$$\begin{array}{lll} \mathcal{X}_u \circ_+ \mathcal{X}_v & = (& X_{u_1},\ldots,X_{u_a},\ldots,X_{u_b},\ldots,X_{u_r}, \\ & & X_{v_1},\ldots,X_{v_c},\ldots,X_{v_d},\ldots,X_{v_\ell}) \\ T(\mathcal{X}_u \circ_+ \mathcal{X}_v) & = \{ & X_{v_c},\ldots,X_{v_d}\} = T(\mathcal{X}_v). \end{array}$$

- The concatenation of $\mathcal{X}_u$ and $\mathcal{X}_v$ which consists of first all bags from $\mathcal{X}_u$ and then all bags from $\mathcal{X}_v$ and where the twin bags of the concatenation are exactly the twin bags of $\mathcal{X}_u$. This is denoted by:

$$\begin{array}{lll} \mathcal{X}_u \circ_- \mathcal{X}_v & = (& X_{u_1},\ldots,X_{u_a},\ldots,X_{u_b},\ldots,X_{u_r}, \\ & & X_{v_1},\ldots,X_{v_c},\ldots,X_{v_d},\ldots,X_{v_\ell}) \\ T(\mathcal{X}_u \circ_- \mathcal{X}_v) & = \{ & X_{v_a},\ldots,X_{v_b}\} = T(\mathcal{X}_u). \end{array}$$

- The concatenation of $\mathcal{X}_u$ and $\mathcal{X}_v$ which consists of first all pendant+ bags from $\mathcal{X}_u$, then all pendant+ bags from $\mathcal{X}_v$, then all twin bags from $\mathcal{X}_u$, then all twin bags from $\mathcal{X}_v$ and all pendant− bags from $\mathcal{X}_u$ followed by all pendant− bags from $\mathcal{X}_v$. Twin bags of this concatenation are of all twin bags of $\mathcal{X}_u$ and $\mathcal{X}_v$. It is denoted by:

$$\begin{array}{llll} \mathcal{X}_u \circ \mathcal{X}_v & = (& X_{u_1},\ldots,X_{u_{a-1}}, & X_{v_1},\ldots,X_{v_{c-1}}, \\ & & X_{u_a},\ldots,X_{u_b}, & X_{v_c},\ldots,X_{v_d}, \\ & & X_{u_{a+1}}\ldots,X_{u_r}, & X_{v_{d+1}},\ldots,X_{v_\ell}) \\ T(\mathcal{X}_u \circ \mathcal{X}_v) & = \{ & X_{v_a},\ldots,X_{v_b},X_{v_c}, & \ldots,X_{v_d}\} = T(\mathcal{X}_u) \cup T(\mathcal{X}_v) \end{array}$$

- The merge of $\mathcal{X}_u$ and $\mathcal{X}_v$ where all twin bags from $\mathcal{X}_v$ get included in the twin bags of $\mathcal{X}_u$. It consists of first all pendant+ bags from $\mathcal{X}_u$, then all pendant+ bags from $\mathcal{X}_v$, then the twin bags of $\mathcal{X}_u$ where every bag is extended by the union of all twin bags of $\mathcal{X}_v$, then all pendant− bags from $\mathcal{X}_u$ followed by all pendant− bags from $\mathcal{X}_v$. The twin bags of this merge are the twin bags of $\mathcal{X}_u$ extended by the union of all twin bags of $\mathcal{X}_v$. This is denoted by:

$$\begin{array}{lll} \mathcal{X}_u \circ_\supset \mathcal{X}_v & = (& X_{u_1},\ldots,X_{u_{a-1}},X_{v_1},\ldots,X_{v_{c-1}}, \\ & & X_{u_a} \cup \left(\bigcup_{c\le i\le d} \mathcal{X}_{v_i}\right), \\ & & X_{u_{a+1}} \cup \left(\bigcup_{c\le i\le d} \mathcal{X}_{v_i}\right),\ldots, \\ & & X_{u_b} \cup \left(\bigcup_{c\le i\le d} \mathcal{X}_{v_i}\right), \\ & & X_{u_{b+1}},\ldots,X_{u_r},X_{v_{d+1}},\ldots,X_{v_\ell}) \\ T(\mathcal{X}_u \circ_\supset \mathcal{X}_v) & = \{ & X_{u_a} \cup \left(\bigcup_{c\le i\le d} \mathcal{X}_{v_i}\right),\ldots,X_{u_b} \cup \left(\bigcup_{c\le i\le d} \mathcal{X}_{v_i}\right)\}. \end{array}$$

We now give the algorithm to compute a directed path-decomposition for a twin-distance-hereditary digraph given by a directed pruning sequence.

Algorithm 1 Computing a directed path-decomposition P of minimal width for a twin-distance-hereditary input digraph G.

Input: directed pruning sequence $S = ((v_1, op_1, v_{a_1}), \ldots, (v_{n-1}, op_{n-1}, v_{a_{n-1}}))$
for $i = 0, \ldots, n-1$ **do** Initialization
  $\mathcal{X}_{v_i} := (\{v_i\})$
  $T(\mathcal{X}_{v_i}) := \{\{v_i\}\}$
  $t(v_i) := 1$
  $w_t(v_i) := 0$
  $w_p(v_i) := 0$
end for
for $i = n-1, \ldots, 1$ **do** Main Loop
  $op := op_i$
  $a := v_{a_i}$
  $v := v_i$
  Switch op:
    $+:$ $w_p(a) := \max\{w_p(a), w_p(v), w_t(v)\}$
      $\mathcal{X}_a := \mathcal{X}_v \circ_+ \mathcal{X}_a$
    $-:$ $w_p(a) := \max\{w_p(a), w_p(v), w_t(v)\}$
      $\mathcal{X}_a := \mathcal{X}_a \circ_- \mathcal{X}_v$
    $\leftarrow:$ $w_t(a) := \max\{w_t(a), w_t(v)\}$
      $w_p(a) := \max\{w_p(a), w_p(v)\}$
      $\mathcal{X}_a := \mathcal{X}_v \circ \mathcal{X}_a$
      $t(a) := t(a) + t(v)$
    $\rightarrow:$ $w_t(a) := \max\{w_t(a), w_t(v)\}$
      $w_p(a) := \max\{w_p(a), w_p(v)\}$
      $\mathcal{X}_a := \mathcal{X}_a \circ \mathcal{X}_v$
      $t(a) := t(a) + t(v)$
    $\diamond:$ $w_t(a) := \max\{w_t(a), w_t(v)\}$
      $w_p(a) := \max\{w_p(a), w_p(v)\}$
      $\mathcal{X}_a := \mathcal{X}_a \circ \mathcal{X}_v$
      $t(a) := t(a) + t(v)$
    $\leftrightarrow:$ $w_t(a) := \min\{w_t(a) + t(v), w_t(v) + t(a)\}$
      $w_p(a) := \max\{w_p(a), w_p(v)\}$
      if $w_t(a) + t(v) \leq w_t(v) + t(a)$ **then**
        $\mathcal{X}_a := \mathcal{X}_a \circ_\supset \mathcal{X}_v$
      else
        $\mathcal{X}_a := \mathcal{X}_v \circ_\supset \mathcal{X}_a$
      end if
      $t(a) := t(a) + t(v)$
end for
Return $\mathcal{X}_{v_0}$, $\max\{w_p(v_0), w_t(v_0)\}$

In the algorithm we use the following variables:

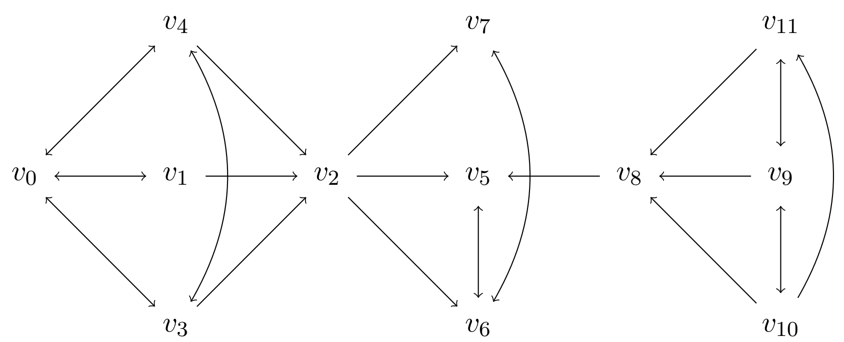
- $\mathcal{X}_{v_i}$ is a minimal directed path-decomposition of a subdigraph of the input graph, which is generated by v_i and following operations in the pruning sequence
- $T(\mathcal{X}_{v_i})$ is the function which gives the twin bags of every $\mathcal{X}_{v_i}$. Please note that it is updated by the previously defined concatenations and merge in every step
- $t(v_i)$ is the number of elements in $T(\mathcal{X}_{v_i})$ in every step of the algorithm
- $w_t(v_i)$ is the width of $\mathcal{X}'_{v_i}$, which is generated by $\mathcal{X}_{v_i}$ by deleting all bags that are not in $T(\mathcal{X}_{v_i})$
- $w_p(v_i)$ is the width of $\mathcal{X}''_{v_i}$, which is generated by $\mathcal{X}_{v_i}$ by deleting all bags that are in $T(\mathcal{X}_{v_i})$

The algorithm returns a directed path-decomposition and the directed path-width of G. A more detailed explanation of the algorithm is in the proof of lemma 5.4.19.

We now give an example how Algorithm 1 works, before we show correctness of the algorithm.

Example 5.4.18. Let G be a directed twin-distance-hereditary graph with the following pruning sequence:

$$\begin{aligned} S \;=\; & ((v_1, \leftrightarrow, v_0), (v_2, -, v_1), (v_3, \leftarrow, v_1), (v_4, \leftrightarrow, v_3), (v_5, -, v_2), (v_6, \leftrightarrow, v_5), \\ & (v_7, \diamond, v_5), (v_8, +, v_5), (v_9, +, v_8), (v_{10}, \leftrightarrow, v_9), (v_{11}, \leftarrow, v_{10})) \end{aligned}$$



Now use Algorithm 1 to find the directed path-width of this graph. First, for all $0 \leq i \leq 11$: Let $t(v_i) = 1$, $w_t(v_i) = 0$, $w_p(v_i) = 0$, $\mathcal{X}_{v_i} = (\{v_i\})$, $T(\mathcal{X}_{v_i}) = \{\{v_i\}\}$.

$$\begin{aligned} (v_{11}, \leftarrow, v_{10}): \quad & w_t(v_{10}) := \max\{w_t(v_{10}), w_t(v_{11})\} = 0 \\ & w_p(v_{10}) := \max\{w_p(v_{10}), w_p(v_{11})\} = 0 \\ & \mathcal{X}_{v_{10}} := \mathcal{X}_{v_{10}} \circ \mathcal{X}_{v_{11}} = (\{v_{10}\}, \{v_{11}\}) \\ & T(\mathcal{X}_{v_{10}}) = \{\{v_{10}\}, \{v_{11}\}\} \\ & t(v_{10}) := t(v_{11}) + t(v_{10}) = 2 \\ (v_{10}, \leftrightarrow, v_9): \quad & w_t(v_9) := \min\{w_t(v_9) + t(v_{10}), w_t(v_{10}) + t(v_9)\} = \min\{2, 1\} = 1 \\ & w_p(v_9) := \max\{w_p(v_9), w_p(v_{10})\} = 0 \\ & \mathcal{X}_{v_9} := \mathcal{X}_{v_{10}} \circ_{\supset} \mathcal{X}_{v_9} = (\{v_9, v_{10}\}, \{v_9, v_{11}\}) \\ & T(\mathcal{X}_{v_9}) = \{\{v_9, v_{10}\}, \{v_9, v_{11}\}\} \end{aligned}$$

$$
\begin{array}{ll}
 & t(v_9) := t(v_9) + t(v_{10}) = 3 \\
(v_9, +, v_8): & w_p(v_8) := \max\{w_p(v_8), w_p(v_9), w_t(v_9)\} = 1 \\
 & \mathcal{X}_{v_8} := \mathcal{X}_{v_9} \circ_+ \mathcal{X}_{v_8} = (\{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}) \\
 & T(\mathcal{X}_{v_8}) = \{\{v_8\}\} \\
(v_8, +, v_5): & w_p(v_5) := \max\{w_p(v_5), w_p(v_8), w_t(v_8)\} = 1 \\
 & \mathcal{X}_{v_5} := \mathcal{X}_{v_8} \circ_+ \mathcal{X}_{v_5} = (\{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}, \{v_5\}) \\
 & T(\mathcal{X}_{v_5}) = \{\{v_5\}\} \\
(v_7, \diamond, v_5): & w_t(v_5) := \max\{w_t(v_5), w_t(v_7)\} = 0 \\
 & w_p(v_5) := \max\{w_p(v_5), w_p(v_7)\} = 1 \\
 & \mathcal{X}_{v_{10}} := \mathcal{X}_{v_{10}} \circ \mathcal{X}_{v_{11}} = (\{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}, \{v_5\}, \{v_7\}) \\
 & T(\mathcal{X}_{v_5}) = \{\{v_5\}, \{v_7\}\} \\
 & t(v_{10}) := t(v_5) + t(v_7) = 2 \\
(v_6, \leftrightarrow, v_5): & w_t(v_5) := \min\{w_t(v_5) + t(v_6), w_t(v_6) + t(v_5)\} = \min\{0+1, 0+2\} = 1 \\
 & w_p(v_6) := \max\{w_p(v_5), w_p(v_6)\} = 1 \\
 & \mathcal{X}_{v_5} := \mathcal{X}_{v_5} \circ_\supset \mathcal{X}_{v_6} = (\{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}, \{v_5, v_6\}, \{v_6, v_7\}) \\
 & T(\mathcal{X}_{v_5}) = \{\{v_5, v_6\}, \{v_6, v_7\}\} \\
 & t(v_5) := t(v_5) + t(v_6) = 3 \\
(v_5, -, v_2): & w_p(v_2) := \max\{w_p(v_2), w_p(v_5), w_t(v_5)\} = 1 \\
 & \mathcal{X}_{v_2} := \mathcal{X}_{v_2} \circ_- \mathcal{X}_{v_2} = (\{v_2\}, \{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}, \{v_5, v_6\}, \{v_6, v_7\}) \\
 & T(\mathcal{X}_{v_5}) = \{\{v_5\}\} \\
(v_4, \leftrightarrow, v_3): & w_t(v_3) := \min\{w_t(v_3) + t(v_4), w_t(v_4) + t(v_3)\} = \min\{0+1, 0+1\} = 1 \\
 & w_p(v_3) := \max\{w_p(v_3), w_p(v_4)\} = 0 \\
 & \mathcal{X}_{v_3} := \mathcal{X}_{v_3} \circ_\supset \mathcal{X}_{v_4} = (\{v_3, v_4\}) \\
 & T(\mathcal{X}_{v_3}) = \{\{v_3, v_4\}\} \\
 & t(v_3) := t(v_3) + t(v_4) = 2 \\
(v_3, \leftarrow, v_1)): & w_t(v_1) := \max\{w_t(v_1), w_t(v_3)\} = 1 \\
 & w_p(v_1) := \max\{w_p(v_1), w_p(v_3)\} = 0 \\
 & \mathcal{X}_{v_1} := \mathcal{X}_{v_1} \circ \mathcal{X}_{v_3} = (\{v_1\}, \{v_3, v_4\}) \\
 & T(\mathcal{X}_{v_1}) = \{\{v_1\}, \{v_3, v_4\}\} \\
 & t(v_1) := t(v_1) + t(v_3) = 3 \\
(v_2, -, v_1): & w_p(v_1) = \max\{w_p(v_1), w_p(v_2), w_t(v_2)\} = 1 \\
 & \mathcal{X}_{v_1} := \mathcal{X}_{v_1} \circ_- \mathcal{X}_{v_2} = (\{v_1\}, \{v_3, v_4\}, \{v_2\}, \{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}, \\
 & \qquad \{v_5, v_6\}, \{v_6, v_7\}) \\
 & T(\mathcal{X}_{v_1}) = \{\{v_1\}, \{v_3, v_4\}\} \\
(v_1, \leftrightarrow, v_0): & w_t(v_0) = \min\{w_t(v_0) + t(v_1), w_t(v_1) + t(v_0)\} = \min\{0+3, 1+1\} = 2 \\
 & w_p(v_0) := \max\{w_p(v_0), w_p(v_1)\} = 1 \\
 & \mathcal{X}_{v_0} := \mathcal{X}_{v_1} \circ_\supset \mathcal{X}_{v_0} = (\{v_0, v_1\}, \{v_0, v_3, v_4\}, \{v_2\}, \{v_9, v_{10}\}, \\
 & \qquad \{v_9, v_{11}\}, \{v_8\}, \{v_5, v_6\}, \{v_6, v_7\}) \\
 & T(\mathcal{X}_{v_0}) = \{\{v_0, v_1\}, \{v_0, v_3, v_4\}\} \\
 & t(v_0) := t(v_0) + t(v_1) = 4
\end{array}
$$

Return $\mathcal{X}_{v_0}$, $\max\{w_t(v_0), w_p(v_0)\} = 2$

The generated directed path-decomposition for digraph G is therefore $\mathcal{X} = \mathcal{X}_{v_0} =$

$(\{v_0, v_1\}, \{v_0, v_3, v_4\}, \{v_2\}, \{v_9, v_{10}\}, \{v_9, v_{11}\}, \{v_8\}, \{v_5, v_6\}, \{v_6, v_7\})$. This path-decomposition has width 2, which equals to the directed path-width of G. Therefore, $\mathcal{X}$ is a minimum and thus, optimum directed path-decomposition for G.

Lemma 5.4.19. *Let G be a twin-distance-hereditary digraph with $|V(G)| = n$ and directed pruning sequence S. Algorithm 1 gives the directed path-width of graph G in time $\mathcal{O}(n)$.*

Proof. Let $S = (s_1, \dots s_{n-1})$ be the directed pruning sequence of G with corresponding vertex-ordering $\sigma = (v_0, \dots, v_{n-1})$. We now use Algorithm 1 to generate a directed path-decomposition for G. As the pruning sequence contains exactly $n-1$ elements and the algorithm passes each element once for the initialization and once again during the algorithm, performing at maximum 3 operations, it works in time $\mathcal{O}(n)$. Remains to show, that Algorithm 1 gives a minimal directed path-decomposition for G.

First we need some definitions. For v_j a vertex of $V(G)$, let $G(v_j)_i$ be the graph consisting of v_j and every vertex that is generated by operations on v_j after step i, which means that $G(v_j)_i$ is created by the pruning sequence $S(v_j)_i$ which contains elements $s_k = (v_k, op_k, v_{a_k})$ with $k \geq i$ and v_{a_k} has been generated by a series of operations by v_j. For $i = n$, this means that $G(v_j)_i = (\{a\}, \emptyset)$. Note that $S(v_0)_1 = S$ and $G(v_0)_1 = G$.

We further say a vertex v_i is a *far twin* of v_j in G if v_i can be created by a series of twin operations from v_j. Formally, this means that there is a series of elements $s_{j_1}, \dots, s_{j_k}$ such that $s_{j_1} = (v_{j_1}, op_{j_1}, v_i), s_{j_2} = (v_{j_2}, op_{j_2}, v_{j_1}), \dots, s_{j_k} = (v_{j_k}, op_{j_k}, v_{j_{k-1}}) = s_j$ where $op_{j_\ell} \in \{\diamond, \leftarrow, \rightarrow, \leftrightarrow\}$ for $1 \leq \ell \leq k$. Let further be every vertex a far twin of itself.

The algorithm works on the pruning sequence starting at the last element. We will now show that after the initialization (step n) and then after every step of the main loop (steps $n-1$ to 1), so for $1 \leq i \leq n$, for every vertex $v_j \in V(G)$ it holds that

1. $\mathcal{X}_{v_j}$ is a minimal directed path-decomposition of $G(v_j)_i$
2. $T(\mathcal{X}_{v_j})$ is the set of all bags in $\mathcal{X}_{v_j}$ containing a far twin of v_j.
3. $t(v_j) = \left|\bigcup\{u \in X \mid X \in T(\mathcal{X}_{v_j})\}\right|$
4. $w_t(v_j)$ is the width of $\mathcal{X}'_{v_j}$, which is generated by $\mathcal{X}_{v_j}$ by deleting all bags that are not in $T(\mathcal{X}_{v_j})$
5. $w_p(v_j)$ is the width of $\mathcal{X}''_{v_j}$, which is generated by $\mathcal{X}_{v_j}$ by deleting all bags that are not in $T(\mathcal{X}_{v_j})$

We show this by induction on the pruning sequence. After the initialization, the statement is true: Let $v_j \in V(G)$. It then holds that $G(v_j)_n$ is the graph that consists only of one vertex v_j. It then holds by the initialization of the algorithm:

1. $\mathcal{X}_{v_j} = (\{v_j\})$ is a minimal directed path-decomposition of $G(v_j)_n$

2. As v_j is the last inserted vertex, there is no vertex generated from v_j by any operations, so $T(\mathcal{X}_v) = \{v\}$ is the set of all bags in $\mathcal{X}_{v_j}$ containing a vertex that is generated by v_j by a series of twin operations including v_j.
3. $t(v_j) = 1 = \left|\bigcup\{u \in X \mid X \in T(\mathcal{X}_{v_j})\}\right|$
4. $w_t(v_j) = 0$ is the width of $\mathcal{X}'_{v_j} = (\{v_j\})$
5. $w_p(v_j) = 0$ is the width of $\mathcal{X}''_{v_j}$, which is an empty decomposition.

Let $1 \leq i < n+1$. We now assume that the statement is true for $i+1$ and show that it is still true for step i. Let $s_i = (v_i, op_i, v_{a_i})$ be the pruning element which is treated in step i of the algorithm. To simplify, we denote $(v_i, op_i, v_{a_i}) =: (v, op, a)$. As it holds that $G(v_j)_i = G(v_j)_{i+1}$ for $v_j \neq a$ and $\mathcal{X}_{v_j}$, $T(v_j)$, $t(v_j)$, $w_t(v_j)$ and $w_p(v_j)$ do not change in this step for $v_j \neq a$, it only remains to show that the statement is true for a after step i. Further, note that $V(G(a)_i) = V(G(v)_{i+1}) \cup V(G(a)_{i+1})$ and $V(G(v)_{i+1} \cap V(G(a)_{i+1}) = \emptyset$. We now consider the different cases of the algorithm:

- $op \in \{+, \ -\}$:
  1. If v is a pendant plus vertex of a, this means by construction rules of a twin-distance-hereditary digraph that every edge between $V(G(v)_{i+1})$ and $V(G(a)_{i+1})$ is an arc from a vertex in $V(G(v)_{i+1})$ to $V(G(a)_{i+1})$, i.e. there is no strong component containing v and a. Then, for creating a directed path-decomposition for $G(a)_i$, it is possible to just concatenate the two path-decompositions of $V(G(v)_{i+1})$ and $V(G(a)_{i+1})$. Therefore, $\mathcal{X}_a := \mathcal{X}_v \circ_+ \mathcal{X}_a$ is a directed path-decomposition of $G(a)_i$. As $G(v)_{i+1}$ and $G(a)_{i+1}$ are induced subdigraphs of $G(a)_i$, this decomposition is minimal. For a pendant minus vertex, the argumentation is equal, though the direction of the arcs and so the order of v and a is the other way round.
  2. By definition of the $\circ_+$ and the $\circ_-$ operator, $T(a)$ does not change in this step. As v is a pendant vertex of a, there is no far twin of a in $G(v)_{i+1}$. As further $V(G(v)_{i+1}) \cap V(G(a)_{i+1}) = \emptyset$ and $T(\mathcal{X}_a)$ contains all bags of the path-decomposition of $G(a)_{i+1}$ containing a far twin of a and as $\mathcal{X}_a$ is a concatenation of the path-decompositions of $G(v)_{i+1}$ and $G(a)_{i+1}$ after step s_i, $T(a)$ still is the set of all bags in $\mathcal{X}_a$ containing a far twin of a.
  3. $t(v)$ does not change in this step. As before this step it holds that $t(a) = |\bigcup\{u \in X \mid X \in T(\mathcal{X}_a)\}|$ by induction and $T(a)$ does not change, this remains true.
  4. $w_t(a)$ does not change in this step. As $T(\mathcal{X}_a)$ does not change, $\mathcal{X}'_a$ does not change either and thus $w_t(a)$ is by induction the width of $\mathcal{X}'_a$.
  5. As $\mathcal{X}_a$ changes but $T(\mathcal{X}_a)$ does not, $w_p(a)$ has to change, too. As $\mathcal{X}_a$ now contains all elements previously included in $\mathcal{X}_a$ and $\mathcal{X}_v$, but only $T(\mathcal{X}_a)$ does not change, $\mathcal{X}''_a$ contains all elements of $\mathcal{X}_v$ and all elements that where previously contained in $\mathcal{X}_v$, but not in $T(\mathcal{X}_v)$. Thus, the width of $\mathcal{X}''_a$ is exactly $\max\{w_p(a), w_p(v), w_t(v)\}$.

- $op \in \{\rightarrow, \leftarrow, \diamond\}$:

  1. By definition of $\rightarrow$, $\leftarrow$ and $\diamond$, for all these options, as in the first case, it holds that there is no strong component containing v and a. So here, as before, it is possible to concatenate the two path-decompositions of $G(a)_{i+1}$ and $G(v)_{i+1}$, where the order is given by the operation. (For $\rightarrow$, it has to be far twins of a before far twins of v, for $\leftarrow$ the other way round and for $\diamond$ both is possible.) As only a and v and their twins matter, we here concatenate in a way such that the created path-decomposition still has the form, that $T(\mathcal{X}_a)$ is placed in a connected part of the path-decomposition. As $G(v)_{i+1}$ and $G(a)_{i+1}$ are induced subdigraphs of $G(a)_i$, this decomposition is minimal.
  2. As v is a twin of a, all far twins of v are also far twins of a. As $V(G(a)_i) = V(G(v)_{i+1}) \cup V(G(a)_{i+1})$ and $T(\mathcal{X}_a) := T(\mathcal{X}_v) \cup T(\mathcal{X}_a)$, $T(\mathcal{X}_a)$ is the set of all bags in $\mathcal{X}_a$ containing a far twin of a.
  3. $t(a) := t(a) + t(v)$ and as $T(\mathcal{X}_a) := T(\mathcal{X}_v) \cup T(\mathcal{X}_a)$ and those two sets are disjoint, $t(a) = |\bigcup\{u \in X \mid X \in T(\mathcal{X}_a)\}|$
  4. $w_t(a) := \max\{w_t(a), w_t(v)\}$ as $\mathcal{X}_a := \mathcal{X}_a \circ \mathcal{X}_v$ and $T(\mathcal{X}_a) := T(\mathcal{X}_v) \cup T(\mathcal{X}_a)$.
  5. $w_p(a) := \max\{w_p(a), w_p(v)\}$ for the same reason as in 4.

- $op = \leftrightarrow$:

  1. The operation $\leftrightarrow$ means, that there is a bidirectional arc in $G(a)_i$ between every far twin of a in $V(G(a)_{i+1})$ and every far twin of v in $V(G(v)_{i+1})$. There are created no edges between any vertices, that are not far twins of a and v. Therefore, to create a directed path-decomposition of $G(a)_i$, only vertices that are far twins of a and v matter, and those are saved in $T(\mathcal{X}_a)$ and $T(\mathcal{X}_v)$ before step i. Every vertex in $G(a)_i$, which is not included in $T(\mathcal{X}_a)$ and $T(\mathcal{X}_v)$ can thus be simply concatenated, regarding the placement before or after the sets $T(\mathcal{X}_a)$ and $T(\mathcal{X}_v)$ in the created path-decomposition. Now the question is, how to unify the part of the path-decomposition containing $T(\mathcal{X}_a)$ and $T(\mathcal{X}_v)$. The graphs generated by the vertex sets $T(\mathcal{X}_a)$ and $T(\mathcal{X}_v)$ are co-graphs, which is easy to see by Theorem 5.4.7. As $\mathcal{X}'_a$ and $\mathcal{X}'_v$ are the path-decompositions of those co-graphs and there are double arcs between all vertices in $T(\mathcal{X}_a)$ and all vertices in $T(\mathcal{X}_v)$, by the same argumentation as in Theorem 1 in [GR18], either $\mathcal{X}'_a \circ_{\supset} \mathcal{X}'_v$ or $\mathcal{X}'_v \circ_{\supset} \mathcal{X}'_a$ give a minimal directed path-decomposition of the induced subdigraph of $G(a)_i$ generated by $T(\mathcal{X}_a) \cup T(\mathcal{X}_v)$. Therefore $\mathcal{X}_a \circ_{\supset} \mathcal{X}_v$ or $\mathcal{X}_v \circ_{\supset} \mathcal{X}_a$ is a minimal directed path-decomposition for the whole graph $G(a)_i$, and the smaller of those two is chosen here to be set in $\mathcal{X}_a$ (see 4).
  2. As a and v are twins, all far twins of v are also far twins of a. All bags including far twins of a or v, have not been concatenated, but been merged

in $\mathcal{X}_a$. Those are exactly the bags, which are set in $T(\mathcal{X}_a)$ in this step by the $\circ_\supset$ operation. So $T(a)$ is the set of all bags in $\mathcal{X}_a$ containing a far twin of a.

3. As all bags, which have been in $T(\mathcal{X}_v)$ and $T(\mathcal{X}_a)$ have been merged in this step and included in the new $T(\mathcal{X}_a)$, and $V(G(a)_{i+1}) \cap V(G(v)_{i+1}) = \emptyset$, all vertices in the bags of the new $T(\mathcal{X}_a)$ now are vertices that have been in $T(\mathcal{X}_v)$ and $T(\mathcal{X}_a)$ before. Therefore, we get that $t(a) := t(a) + t(v) = |\bigcup\{u \in X \mid X \in T(\mathcal{X}_a)\}|$.
4. As in 1, we here also use the argumentation of Theorem 1 in [GR18]. As the graphs generated by the old sets $T(\mathcal{X}_a)$ and $T(\mathcal{X}_v)$ are co-graphs, $\mathcal{X}'_a \circ_\supset \mathcal{X}'_v$ or $\mathcal{X}'_v \circ_\supset \mathcal{X}'_a$ give a minimal directed path-decomposition of the induced subdigraph of $G(a)_i$ and their size is $t(a) + w_t(v)$ or $w_t(a) + t(v)$. The algorithm chooses the smaller value by the if-statement and then the associated decomposition and thus, $w_t(a) := \min\{w_t(a)+t(v), w_t(v)+t(a)\}$ is the width of $\mathcal{X}'_a = \{a\}$.
5. Here, $\mathcal{X}''_a$ contains exactly all elements, which has been in $\mathcal{X}''_a$ previously and all which of $\mathcal{X}''_v$. Therefore, its width is $\max\{w_p(a), w_p(v)\}$.

By this induction, it finally holds that after the last step of the algorithm, when s_1 has been processed, that $\mathcal{X}_a = \mathcal{X}_{v_0}$ is a minimal directed path-decomposition of $G(v_0)_1 = G$. Further, as all elements of $\mathcal{X}_0$ are either contained in $\mathcal{X}'_0$ or in $\mathcal{X}''_0$, $\max\{w_p(v_0), w_t(v_0)\}$ gives the width of the decomposition $\mathcal{X}_{v_0}$ which is the directed path-width of G.

Please note that the Operation $\circ_\supset$ can not be performed in constant time. Thus, Algorithm 1, as it is presented above, does not work in linear time. But by modifying it such that it does not create a directed path-decomposition but only the directed path-width (which means deleting lines 15, 17, 20, 24, 28, 32-36), the algorithm takes linear time and does compute the directed path-width of the input graph G.

So, the directed path-width of G is computable in linear time. □

As obtaining a minimal directed path-decomposition of G in time $\mathcal{O}(n)$ also allows to get the directed path-width in the same time, Theorem 5.4.17 follows directly.

Directed Clique-width of Twin-DH Digraphs

We now consider the already mentioned parameter directed clique-width. It differs from the previously mentioned parameters as instead of representing the size of strong components in some way, it describes the number of different neighbourhoods. Especially for bioriented cliques, the above mentioned parameters are infinitely large whereas directed clique-width is linear.

Undirected clique-width was introduced by [CO00] and is defined correspondingly. In the undirected case, co-graphs are exactly the graphs of clique-width at most 2 and distance-hereditary graphs have clique-width at most 3, which leads to the idea of regarding directed clique-width on twin-dh digraphs.

Theorem 5.4.20. *Every twin-dh digraph has directed clique-width at most* 3.

Proof. We show a construction for a directed clique-width 3-expression for every $G = (V, E) \in DDH$ and then argue, why this is best possible. The method we use is closely related to the undirected case: Distance-hereditary graphs have clique-width at most 3, see [GR00]. Let $G \in$ DDH and $S = (s_1, \ldots, s_n)$ be a directed pruning sequence creating G. We give an algorithm to construct a 3-expression traversing S starting with the last element of the sequence. The idea for computing this expression is to use three labels $1, 2$ and 3 as follows: After every step of the algorithm, expressions which are already constructed consist of vertices labeled with 1 and 3, where 1 means that the vertex has not been finally treated and possibly has edges to other vertices not inserted yet, whereas for vertices labeled by 3 all incident edges have already been considered. The label 2 is only used as a working label. For initialization, let $X_v = \bullet_1$ for every vertex $v \in V$. As this is a 1-expression, it is also a 3-expression. In the following let now $s_i = (v_i, op_i, a_i)$, for simplification denoted by (v, op, a), be the currently treated element of S and X_v and X_a be the 3-expressions which exists by induction for v and a. Then, we get a 3-expression by the following rules depending on the operation op.

(1) $op = +: \quad X_a := \rho_{2\to3}(\alpha_{2,1}(\rho_{1\to2}(X_v) \oplus X_a))$

(2) $op = -: \quad X_a := \rho_{2\to3}(\alpha_{1,2}(\rho_{1\to2}(X_v) \oplus X_a))$

(3) $op = \diamond: \quad X_a := X_v \oplus X_a$

(4) $op = \rightarrow: \quad X_a := \rho_{2\to1}(\alpha_{2,1}(\rho_{1\to2}(X_v) \oplus X_a))$

(5) $op = \leftarrow: \quad X_a := \rho_{2\to1}(\alpha_{1,2}(\rho_{1\to2}(X_v) \oplus X_a))$

(6) $op = \leftrightarrow: \quad X_a := \rho_{2\to1}(\alpha_{1,2}(\alpha_{2,1}(\rho_{1\to2}(X_v) \oplus X_a)))$

To prove correctness we first need some definitions. For w a vertex of $V(G)$, let $G(w)_i$ be the graph consisting of w and every vertex that is generated by operations on w after step i, which means that $G(w)_i$ is created by the pruning sequence $S(w)_i$ which contains elements $s_k = (v_k, op_k, v_{a_k})$ with $k \geq i$ and v_{a_k} has been generated by a series of operations by w. For $i = n$, this means that $G(w)_i = (\{w\}, \emptyset)$. Note that for element v_0, which is the first anchor in the pruning sequence, i.e. $s_1 = (v_1, op_1, v_0)$, it holds that $S(v_0)_1 = S$ and $G(v_0)_1 = G$.

We then show by induction that at any step i with $n \geq i \geq 0$ of the algorithm, it holds that X_w is a 3-expression of $G(w)_i$ for all vertices $w \in V$. We further assume that every X_w contains only vertices labeled by 1 and 3, where the vertices labeled by 1 are exactly those, which are created by a series of twin operations on w (including w itself). To simplify, we call such a vertex a *far twin* in the following.

After the initialization, it is easy to see for $i = n$ that for all $w \in V$, $X_w = \bullet_1$ is a 3-expression of $G(w)_i = (\{w\}, \emptyset)$. Obviously, the only vertex in $G(w)_i$ is w which is labeled by 1.

Consider now step i. By induction we know that X_a is a 3-expression of $G(a)_{i+1}$ where far twins of a are labeled by 1 and all other vertices are labeled by 3. Further, X_v is a 3-expression of $G(v)_{i+1}$ where all far twins of v are labeled by 1 and all other vertices are labeled by 3. We now show that after step i it holds that X_a is a 3-expression of $G(a)_i$. Therefore, we consider element $s_i =: (v, op, a)$ in step i.

(1) As v is a pendant plus vertex of a, there exist edges from every far twin of v to every far twin of a. By $\rho_{1\to 2}(X_v)$ we relabel every vertex in X_v which is labeled by 1, i.e. every far twin of v with 2. We join this expression with X_a and add edges from all labels 2 to 1, which inserts all edges created by the pendant plus relation of v to a. As v is a pendant plus vertex of a, all far twins of a can not be far twins of v and thus, we relabel these vertices from 2 to 3. Now, the newly created X_a is a 3-expression of $G(a)_i$ consisting only of labels 1 and 3, where the vertices labeled by 1 are exactly the far twins of a.

(2) Analogously to (1).

(3) As v is a false twin of a, no new edges are inserted by this operation and further all far twins of v are also far twins of a. Therefore, we only need to join expressions X_v and X_a to create an expression for $G(a)_i$ where every far twin of a is labeled by 1 and every other vertex is labeled by 3.

(4) As v is a true in-twin of a, like in (1), there exist edges from every far twin of v to every far twin of a. We therefore use the same method to join the expressions X_v and X_a and create edges between them. But unlike in (1), every far twin of v is a far twin of a. Therefore, we relabel the far twins of v from 2 to 1, to obtain a 3-expression for $G(a)_i$ in which exactly all far twins of a are labeled by 1.

(5) Analogously to (4).

(6) Is very similar to (4) and (5), with the only difference that we need edges from every far twin of v to every far twin of a and the other way round. Therefore, we insert edges from labels 2 to 1 and from labels 1 to 2, before relabeling, to obtain a 3-expression for $G(a)_i$ in which exactly all far twins of a are labeled by 1.

Further, the directed clique-width of a twin-dh digraph has to be at least 3, as can be seen by the following counterexample: The $\overrightarrow{P_3}$, which means a directed path of 3 vertices, is twin-distance-hereditary, but it is not expressible by a 2-expression.

□

In [CMR00], Courcelle et al. proved that every problem, which is describable in monadic second order logic, is computable in polynomial time on clique-width bounded graphs. As the directed clique-width of a graph is always greater or equal the clique-width of the underlying undirected graph, this result can be extended to directed clique-width. It therefore holds that:

Corollary 5.4.21. *Let G be a twin-dh digraph. Then every graph problem, which is describable in MSO_1 logic, is computable in polynomial time on G.*

5.4.4 Further Problems on directed twin-dh graphs

From the result in [GKR21a] we can follow, that the r-dichromatic number problem can be solved in polynomial time on twin-dh digraphs.

By [GWY16] we can solve the problems Directed Hamiltonian Path, Directed Hamiltonian Cycle, Directed Cut, and Regular Subdigraph using an XP-algorithm w.r.t. the parameter directed NLC-width in polynomial time. Directed NLC-width is a digraph parameter which is closely related to directed clique-width, since we can transform every directed clique-width k-expression into an equivalent directed NLC-width k-expression, see [GWY16]. Thus, directed twin-dh graphs have bounded directed NLC-width and we can solve the above mentioned problems in polynomial time on this class.

5.4.5 Conclusion and Outlook

In this section, we introduce directed twin-distance-hereditary graphs, which are developed by a generalization of the recursive definition for undirected distance-hereditary graphs. The class of twin-dh digraphs is a superclass of directed co-graphs, defined in [CP06] and when excluding the bioriented true twin operation, it is a subclass of distance-hereditary digraphs, defined in [LS10]. We characterize this class by forbidden induced subdigraphs. Further, we show that the class is a subclass of extended directed co-graphs which allows us to deduce interesting results. However, due to the unbounded directed clique-width of extended directed co-graphs, twin-dh digraphs exhibit properties which allow supplemental results such that an investigation of this class is advisable. We show interesting properties of the class such as that every strong component is a directed co-graph. This is helpful in the computation of several problems. For future work it might be interesting to investigate if problems which are solvable on undirected distance-hereditary graphs, as e.g. the (directed) Steiner tree problem can also be solved on twin-dh digraphs.

Moreover, we show that several directed width parameters, namely directed path-width, directed tree-width, DAG-width and cycle rank can be computed in linear time on directed twin-distance-hereditary graphs. From the associated proof (as well as from the fact that this twin-dh digraphs are a subclass of extended directed co-graphs) further the equality of all these parameters follows.

Further, we can conclude by our results and [GKR21b] that for twin-dh digraphs, as for directed co-graphs, Kelly-width can be bounded by DAG-width. Due to [HK08, Conjecture 30], [AKK+15], and [BJG18, Section 9.2.5] this remains open for general digraphs and is related to one of the biggest open problems in graph searching, namely whether the monotonicity costs for Kelly- and DAG-width games are bounded. In previous sections we could show, that the equivalence of those two parameters is given on directed co-graphs. In this section we can extend this result to their superclass of twin-dh digraphs.

Like in the undirected case, every twin-dh digraph has directed clique-width at most 3, though not every digraph of directed clique-width 3 is a twin-dh digraph. From that we can conclude several interesting results, since there are many NP-hard problems which are solvable on digraphs of bounded directed clique-width.

It would be interesting for future work to consider other superclasses of twin-dh digraphs and whether it is still possible to find efficient algorithms to compute several graph parameters on these classes and at which point it becomes NP-hard.

5.5 Sequence Digraphs

We now present the set of sequence digraphs and an algorithm to obtain directed path-width on this graph class.

A *sequence digraph* is defined by a set $Q = \{q_1, \dots, q_k\}$ of k sequences $q_i = (b_{i,1}, \dots, b_{i,n_i})$, $1 \leq i \leq k$. Further there is a function t which assigns to every item $b_{i,j}$ a type $t(b_{i,j})$. The sequence digraph $g(Q) = (V, A)$ for the set Q has a vertex for every type and an arc $(u, v) \in A$ if and only if there is some sequence q_i in Q where an item of type u is on the left of some item of type v. The set of all sequence digraphs which can be defined by sets Q on at most k sequences that together contain at most ℓ items of each type is denoted by $S_{k,\ell}$.

We show in Theorem 5.5.17 that $S_{1,1}$ is equal to the well known class of transitive tournaments. Since only the first and the last item of each type in every $q_i \in Q$ are important for the arcs in the corresponding digraph all classes $S_{1,\ell}$, $\ell \geq 2$ are equal. We show in Theorem 5.5.25 that $S_{1,2}$ is equal to the set of semicomplete $\{\text{co-}(2\overrightarrow{P_2}), \overrightarrow{C_3}, D_3\}$-free digraphs (cf. Table 5.3 for the digraphs). By our Proposition 5.5.19 set $S_{k,1}$ can be characterized by only three forbidden subdigraphs. It is also the class of disjoint unions of k transitive tournaments.

Considering the directed path-width problem on sequence digraphs, we get some remarkable results. We show that for digraphs defined by $k = 1$ sequence the directed path-width can be computed in polynomial time. Further we show that for sets Q of sequences of bounded length, of bounded distribution of the items of every type onto the sequences, or bounded number of items of every type computing the directed path-width of $g(Q)$ is NP-hard. We show that for a fixed number k of sequences the directed path-width is computable in polynomial time. Therefore in Theorem 5.5.37 we introduce an algorithm which computes the directed path-width of a digraph which is given by a set of k sequences in time $\mathcal{O}(k \cdot (1 + \max_{1 \leq i \leq k} n_i)^k)$. The main idea is to discover an optimal directed path-decomposition by scanning the k sequences left-to-right and keeping in a state the numbers of scanned items of every sequence and a certain number of active types.

From a parameterized point of view our solution leads to an XP-algorithm w.r.t. parameter k. While the existence of FPT-algorithms for computing directed path-width is open up to now, there are further XP-algorithms for the directed path-width problem for some digraph $G = (V, A)$. The directed path-width can be computed in time $\mathcal{O}(|A|\cdot|V|^{2\text{d-pw}(G)}/(\text{d-pw}(G)-1)!)$ by [KKK+16] and in time $\mathcal{O}(\text{d-pw}(G)\cdot|A|\cdot|V|^{2\text{d-pw}(G)})$ by [Nag12]. Further in [KKT15] it is shown how to decide whether the directed path-width of an ℓ-semicomplete digraph is at most w in time $(\ell+2w+1)^{2w}\cdot|V|^{\mathcal{O}(1)}$. All these algorithms are exponential in the directed path-width of the input digraph while our algorithm is exponential within the number of sequences. Thus our result improves theses algorithms for digraphs of large directed path-width which can be decomposed by a small number of sequences (see Table 5.2 for examples). Furthermore the directed path-width can be computed in time $3^{\tau(und(G))}\cdot|V|^{\mathcal{O}(1)}$, where $\tau(und(G))$ denotes the vertex cover number of the underlying undirected graph of G, by [Kob15]. Thus our result also improves this algorithm for digraphs of large $\tau(und(G))$ which can be decomposed by a small number of sequences (see Table 5.2 for examples).

digraphs $G=(V,A)$, $n=\|V\|$	d-pw(G)	$\tau(und(G))$	k	ℓ
transitive tournaments	0	$n-1$	1	1
union of k' transitive tournaments	0	$(\sum_{i=1}^{k'} n_i)-k'$	k'	1
bidirectional complete digraphs $\overleftrightarrow{K_n}$	$n-1$	$n-1$	1	2
semicomplete $\{\overrightarrow{C_3}, D_0, D_3\}$-free	$[0,n-1]$	$n-1$	1	2
semicomplete $\{\text{co-}(2\overrightarrow{P_2}), \overrightarrow{C_3}, D_3\}$-free	$[0,n-1]$	$n-1$	1	2
union of k' semicomplete $\{\text{co-}(2\overrightarrow{P_2}), \overrightarrow{C_3}, D_3\}$-free	$[0,n-1]$	$(\sum_{i=1}^{k'} n_i)-k'$	k'	$2k'$

Table 5.2: Values of parameters within XP-algorithms for directed path-width.

Please note that this section is taken from [GRR18].

5.5.1 From Sequences to Digraphs

Let $Q=\{q_1,\dots,q_k\}$ be a set of k sequences. Every sequence $q_i=(b_{i,1},\dots,b_{i,n_i})$ consists of a number n_i of items, such that all $n=\sum_{i=1}^{k} n_i$ items are pairwise distinct. Further there is a function t which assigns to every item $b_{i,j}$ a type $t(b_{i,j})$. The set of all types of the items in some sequence q_i is denoted by $\mathit{types}(q_i)=\{t(b)\mid b\in q_i\}$. For a set of sequences $Q=\{q_1,\dots,q_k\}$ we denote $\mathit{types}(Q)=\mathit{types}(q_1)\cup\dots\cup\mathit{types}(q_k)$. For some sequence $q_\ell=(b_{\ell,1},\dots,b_{\ell,n_\ell})$ we say item $b_{\ell,i}$ is *on the left of* item $b_{\ell,j}$ in sequence q_ℓ if $i<j$. Item $b_{\ell,i}$ is on the *position* i in sequence q_ℓ, since there are $i-1$ items on the left of $b_{\ell,i}$ in sequence q_ℓ.

In order to *insert* a new item b on a position j in sequence q_i we first move all items on positions $j'\geq j$ to position $j'+1$ starting at the rightmost position n_i and then we insert b at position j. In order to *remove* an existing item b at a position j in sequence q_i we move all items from positions $j'\geq j+1$ to position $j'-1$ starting at position $j+1$.

We consider the distribution of the items of a type t onto the sequences by

$$d_Q(t)=|\{q\in Q\mid t\in \mathit{types}(q)\}| \qquad \text{and} \qquad d_Q=\max_{t\in \mathit{types}(Q)} d_Q(t).$$

For the number of items for type t within the sequences we define

$$c_Q(t) = \sum_{q \in Q} |\{b \in q \mid t(b) = t\}| \qquad \text{and} \qquad c_Q = \max_{t \in types(Q)} c_Q(t).$$

Obviously it holds $d_Q \leq k$ and $1 \leq d_Q \leq c_Q \leq n$.

The *sequence digraph* $g(Q) = (V, A)$ for a set $Q = \{q_1, \ldots, q_k\}$ has a vertex for every type, i.e. $V = types(Q)$ and an arc $(u, v) \in A$ if and only if there is some sequence q_i in Q where an item of type u is on the left of an item of type v. More formally, there is an arc $(u, v) \in A$ if and only if there is some sequence q_i in Q, such that there are two items $b_{i,j}$ and $b_{i,j'}$ such that (1) $1 \leq j < j' \leq n_i$, (2) $t(b_{i,j}) = u$, (3) $t(b_{i,j'}) = v$, and (4) $u \neq v$.

Sequence digraphs have successfully been applied in order to model the stacking process of bins from conveyor belts onto pallets with respect to customer orders, which is an important task in palletizing systems used in centralized distribution centers [GRW16]. In our examples we will use type identifications instead of item identifications to represent a sequence $q_i \in Q$. For r not necessarily distinct types $t_1, \ldots, t_r$ let $[t_1, \ldots, t_r]$ denote some sequence $q_i = (b_{i,1}, \ldots, b_{i,r})$ of r pairwise distinct items, such that $t(b_{i,j}) = t_j$ for $j = 1, \ldots, r$. We use this notation for sets of sequences as well.

Example 5.5.1. Fig. 5.10 shows the sequence digraph $g(Q)$ for $Q = \{q_1, q_2, q_3\}$ with sequences $q_1 = [a, a, d, e, d]$, $q_2 = [c, b, b, d]$, and $q_3 = [c, c, d, e, d]$.

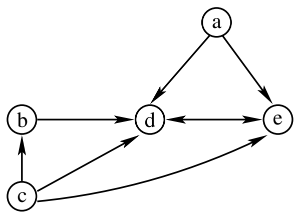


Figure 5.10: Sequence digraph $g(Q)$ of Example 5.5.1.

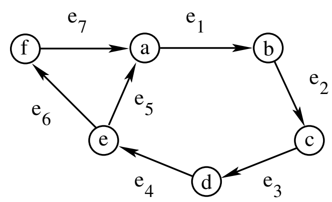


Figure 5.11: Digraph G of Example 5.5.5.

Next we give results in order to compute the sequence digraph $g(Q)$ and also its complement digraph co-$(g(Q)) = \overline{g(Q)}$.[4] Therefore we define the position of the first item in some sequence $q_i \in Q$ of some type $t \in types(Q)$ by $first(q_i, t)$ and the position of the last item of type t in sequence q_i by $last(q_i, t)$. For technical reasons, if there is no item for type t contained in sequence q_i, then we define $first(q_i, t) = n_i + 1$ and $last(q_i, t) = 0$.

Lemma 5.5.2. *Let $Q = \{q_1\}$ be a set of one sequence, $g(Q) = (V, A)$ the sequence digraph, co-$(g(Q)) = (V, A^c)$ its complement digraph, and $u \neq v$, $u, v \in V$.*

[4]Please note that in this section, to improve readability, for a graph G we will use the term co-G for the complement graph instead of $\overline{G}$.

1. *There is an arc* $(u, v) \in A$*, if and only if* $first(q_1, u) < last(q_1, v)$.

2. *There is an arc* $(u, v) \in A^c$*, if and only if* $last(q_1, v) < first(q_1, u)$.

3. *If* $(u, v) \in A^c$*, then* $(v, u) \in A$.

4. *There is an arc* $(u, v) \in A$ *and an arc* $(v, u) \in A^c$*, if and only if* $last(q_1, u) < first(q_1, v)$.

Lemma 5.5.3. *Let* $Q = \{q_1, \dots, q_k\}$ *be a set of* k *sequences,* $g(Q) = (V, A)$ *the sequence digraph, co-*$(g(Q)) = (V, A^c)$ *its complement digraph, and* $u \neq v$*,* $u, v \in V$.

1. *There is an arc* $(u, v) \in A$*, if and only if there is some* $q_i \in Q$ *such that* $first(q_i, u) < last(q_i, v)$.

2. *There is an arc* $(u, v) \in A^c$*, if and only if for every* $q_i \in Q$ *we have* $last(q_i, v) < first(q_i, u)$.

By Lemma 5.5.3(1) only the first and the last item of each type in every $q_i \in Q$ are important for the arcs in the corresponding digraph. Let $M(q_i)$ be the subsequence of q_i which is obtained from q_i by removing all except the first and last item for each type and $M(Q) = \{M(q_1), \dots, M(q_k)\}$.

Observation 5.5.4. Let $Q = \{q_1, \dots, q_k\}$ be a set of k sequences, then $g(Q) = g(M(Q))$.

5.5.2 From Digraphs to Sequences

Let $G = (V, A)$ be some digraph and $A = \{a_1, \dots, a_\ell\}$ its arc set. The *sequence system* $q(G) = \{q_1, \dots, q_\ell\}$ for G is defined as follows. (1) There are 2ℓ items $b_{1,1}, b_{1,2}, \dots, b_{\ell,1}, b_{\ell,2}$. (2) Sequence $q_i = (b_{i,1}, b_{i,2})$ for $1 \leq i \leq \ell$. (3) The type of item $b_{i,1}$ is the first vertex of arc a_i and the type of item $b_{i,2}$ is the second vertex of arc a_i for $1 \leq i \leq \ell$. Thus $types(q(G)) = V$.

Example 5.5.5 (Sequence System). For the digraph G of Fig. 5.11 the corresponding sequence system is $q(G) = \{q_1, q_2, q_3, q_4, q_5, q_6, q_7\}$, where $q_1 = [a, b]$, $q_2 = [b, c]$, $q_3 = [c, d]$, $q_4 = [d, e]$, $q_5 = [e, a]$, $q_6 = [e, f]$, $q_7 = [f, a]$.

By the definition of sequence systems and sequence digraphs we obtain the following result.

Observation 5.5.6. For every digraph G it holds $G = g(q(G))$.

Lemma 5.5.7. *For every digraph* $G = (V, A)$ *with underlying undirected graph* $und(G) = (V, E)$ *there is a set* Q *of at most* $|E|$ *sequences such that* $G = g(Q)$.

There are digraphs which even can be defined by one sequence (see Theorem 5.5.25 for a complete characterization) and there are digraphs for which $|E|$ sequences are really necessary (see Lemma 5.5.13). For digraphs of bounded vertex degree the sequence system $Q = q(G)$ leads to sets whose distribution and number of items of each type can be bounded as follows.

Lemma 5.5.8. *For every digraph $G = (V, A)$ where $\max\{\Delta^-(G), \Delta^+(G)\} \leq d$ there is a set Q with $d_Q \leq 2d$ and $c_Q \leq 2d$ such that $G = g(Q)$.*

In case of complete bioriented digraphs, i.e. we have none or both arcs between any pair of vertices, we can improve the latter bounds.

Lemma 5.5.9. *For every complete bioriented digraph $G = (V, A)$ such that $\max\{\Delta^-(G), \Delta^+(G)\} \leq d$ there is a set Q with $d_Q \leq d$ and $c_Q \leq 2d$ (for $d \geq 2$ even $c_Q \leq 2d - 1$) such that $G = g(Q)$.*

5.5.3 Properties of Sequence Digraphs

Graph Classes and their Relations

We define $S_{k,\ell}$ to be the set of all sequence digraphs defined by sets Q on at most k sequences that contain at most ℓ items of each type in *types*(Q). By Observation 5.5.4 and Lemma 5.5.7 we obtain the following bounds.

Corollary 5.5.10. *Let Q be a set on k sequences and $g(Q) = (V, A) \in S_{k,\ell}$ the defined graph with und$(g(Q)) = (V, E)$. Then we can assume that $1 \leq \ell \leq 2k$ and $1 \leq k \leq |E|$.*

Lemma 5.5.11. *Let $\ell \geq 1$ and $G \in S_{1,\ell}$ be defined by $Q = \{q_1\}$, then $g(Q)$ is semicomplete and graph und$(g(Q))$ is the complete graph on $|types(Q)|$ vertices.*

Next we consider the relations of the defined classes for $k = 1$ sequence. Since $S_{1,1}$ contains only digraphs with exactly one arc between every pair of vertices (cf. Theorem 5.5.17) and $S_{1,\ell}$ for $\ell \geq 2$ contains all bidirectional complete digraphs we know that $S_{1,1} \subsetneq S_{1,\ell}$ for $\ell \geq 2$. Further by $S_{k,\ell} \subseteq S_{k,\ell+1}$ and Observation 5.5.4 it follows that all classes $S_{1,\ell}$ for $\ell \geq 2$ are equal.

Corollary 5.5.12. *For $\ell \geq 2$ the following inclusions hold true.*

$$S_{1,1} \subsetneq S_{1,2} = \ldots = S_{1,\ell}$$

Lemma 5.5.13. *Let $G = (V, E)$ be a triangle free graph, i.e. a C_3-free graph, with $|E| \geq 2$, such that $\Delta(G) = \ell$ and let $G' = (V, A)$ be an orientation of G. Then for $k = |E|$ it holds that $G' \in S_{k,\ell}$ but for $k' < k$ or $\ell' < \ell$ it holds that $G' \notin S_{k',\ell'}$.*

Since for every $k \geq 2$ and every $\ell = 2, \ldots, k$ there is a tree T on k edges and $\Delta(T) = \ell$ we know by Lemma 5.5.13 that for $k \geq 2$ and $\ell = 2, \ldots, k$ it holds $S_{k,\ell-1} \subsetneq S_{k,\ell}$. Further by Observation 5.5.4 we know that for $k \geq 2$ and $\ell \geq 2k$ it holds $S_{k,\ell} = S_{k,\ell+1}$.

Corollary 5.5.14. *For $k \geq 2$ the following inclusions hold true.*

$$S_{k,1} \subsetneq S_{k,2} \subsetneq \ldots \subsetneq S_{k,k} \subseteq S_{k,k+1} \subseteq \ldots \subseteq S_{k,2k} = S_{k,2k+1} = \ldots$$

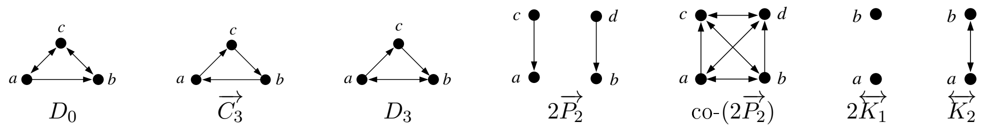


Table 5.3: Special digraphs.

Lemma 5.5.15. *Let $G \in S_{k,\ell}$, then for every induced subdigraph H of G it holds $H \in S_{k,\ell}$.*

Graph classes which are closed under taking induced subgraphs are called *hereditary*. Hereditary graph classes are exactly those classes which can be defined by forbidden induced subgraphs.

Characterizations of Sequence Digraphs for $k = 1$ or $\ell = 1$

In this section we show a finite set of forbidden induced subgraphs for all classes $S_{k,\ell}$ where $\ell = 1$ and for all classes where $k = 1$. These characterizations lead to polynomial time recognition algorithms for the corresponding graph classes. Furthermore we give characterizations in terms of special tournaments and conditions for the complement digraph.

Digraphs for $k = 1$ and $\ell = 1$ A digraph $G = (V, A)$ is called *transitive* if for every pair $(u, v) \in A$ and $(v, w) \in A$ of arcs with $u \neq w$ the arc (u, w) also belongs to A.

Lemma 5.5.16. *Every digraph in $S_{1,1}$ is transitive.*

For a digraph G and an integer d let dG be the disjoint union of d copies of G.

Theorem 5.5.17. *For every digraph G the following statements are equivalent.*

1. *$G \in S_{1,1}$*
2. *G is a transitive tournament.*
3. *G is an acyclic tournament.*
4. *G is a $\overrightarrow{C_3}$-free tournament.*
5. *G is a tournament with exactly one Hamiltonian path.*
6. *G is a tournament and every vertex in G has a different outdegree.*
7. *G is $\{2\overleftrightarrow{K_1}, \overleftrightarrow{K_2}, \overrightarrow{C_3}\}$-free.*
8. *$G \in \{(\{v\}, \emptyset)\} \cup \{(\overrightarrow{P_n})^{n-1} \mid n \geq 2\}$, i.e. G is the $(n-1)$-th power of a directed path $\overrightarrow{P_n}$.*

Proof. The equivalence of (2) – (6) is known from [Gou12, Chapter 9]. (1) ⇒ (2) By Lemma 5.5.16 every digraph $G \in S_{1,1}$ is transitive and by definition of $S_{1,1}$ digraph G is a tournament. (3) ⇒ (1) Every acyclic digraph G has a source, i.e. a vertex v_1 of indegree 0, see [BJG09]. Since G is a tournament there is an arc (v_1, v) for every vertex v of G, i.e. v_1 is an out-dominating vertex. By removing v_1 from G, we obtain a transitive tournament G^1 which leads to an out-dominating vertex v_2. By removing v_2 from G^1, we obtain a transitive tournament G^2 which leads to an out-dominating vertex v_3 and so on. The sequence $[v_1, v_2, \ldots, v_n]$ shows that $G \in S_{1,1}$. (4) ⇔ (7) and (1) ⇔ (8) can be easily verified. □

By part (3) ⇒ (1) of the proof of Theorem 5.5.17 we have shown the next result.

Proposition 5.5.18. *Let $G = (V, A) \in S_{1,1}$, then a sequence q, such that $G = g(\{q\})$ can be found in time $\mathcal{O}(|V| + |A|)$.*

Sequence Digraphs for $\ell = 1$ The sequence digraph $g(Q) = (V, A)$ for a set $Q = \{q_1, \ldots, q_k\}$ can be obtained by the union of $g(\{q_i\}) = (V_i, A_i)$, $1 \leq i \leq k$ by $V = \cup_{i=1}^{k} V_i$ and $A = \cup_{i=1}^{k} A_i$. Since for digraphs in $S_{k,1}$ the vertex sets $V_i = \mathit{types}(q_i)$ are disjoint, all properties of Theorem 5.5.17 can be generalized to $k \geq 1$ sequences. Some of them are given next.

Proposition 5.5.19. *For every digraph G and every integer $k \geq 1$ the following statements are equivalent.*

1. *$G \in S_{k,1}$.*
2. *G is the disjoint union of k digraphs from $S_{1,1}$.*
3. *G is $\{(k+1)\overleftrightarrow{K_1}, \overleftrightarrow{K_2}, \overrightarrow{C_3}\}$-free.*

By Proposition 5.5.18 and Proposition 5.5.19 we have shown the next result.

Proposition 5.5.20. *Let $G = (V, A) \in S_{k,1}$, then a set Q on k sequences, such that $G = g(Q)$ can be found in time $\mathcal{O}(|V| + |A|)$.*

Sequence Digraphs for $k = 1$ The digraph D_0 in Table 5.3 is not transitive, since it has among others the arcs (b, c) and (c, a) but not the arc (b, a). Further D_0 belongs to the set $S_{1,2}$, since it can be defined by set $Q = \{q_1\}$ of one sequence $q_1 = [c, a, b, c]$. Thus for $\ell \geq 2$ items for each type even one sequence can define digraphs which are not transitive. A digraph $G = (V, A)$ is called *quasi transitive* if for every pair $(u, v) \in A$ and $(v, w) \in A$ of arcs with $u \neq w$ there is at least one arc between u and w in A. Since every semicomplete digraph is quasi transitive, by Lemma 5.5.11 every digraph in $S_{1,\ell}$, $\ell \geq 1$, is quasi transitive.

To show characterizations for the class $S_{1,2}$ we next give some lemmas.

Lemma 5.5.21. *Let $\ell \geq 1$ and $G \in S_{1,\ell}$, then its complement digraph co-G is transitive.*

Lemma 5.5.22. *Let $\ell \geq 1$ and $G \in S_{1,\ell}$, then its complement digraph co-G is $2\overrightarrow{P_2}$-free.*

Lemma 5.5.23. *Let G be a semicomplete $\{\overrightarrow{C_3}, D_3\}$-free digraph on n vertices, then G has a vertex v such that outdegree$(v) = n-1$ and a vertex v' such that indegree$(v') = n-1$.*

Lemma 5.5.24. *Every semicomplete $\{\overrightarrow{C_3}, D_3\}$-free digraph has a spanning transitive tournament subdigraph.*

These results allow us to show the following characterizations. Since we use several forbidden induced subdigraphs the semicompleteness is expressed by excluding $2\overleftrightarrow{K_1}$ (see Table 5.3 for the special digraphs).

Theorem 5.5.25. *For every digraph G the following statements are equivalent.*

1. $G \in S_{1,2}$
2. $G \in S_{1,\ell}$ *for some* $\ell \geq 2$
3. *co-G is transitive, co-G is $2\overrightarrow{P_2}$-free, and G has a spanning transitive tournament subdigraph.*
4. *G is $\{$co-$(2\overrightarrow{P_2}), 2\overleftrightarrow{K_1}, \overrightarrow{C_3}, D_3\}$-free.*

Proof. (2) $\Rightarrow$ (1) By Corollary 5.5.12. (1) $\Rightarrow$ (4) co-$(2\overrightarrow{P_2}), 2\overleftrightarrow{K_1}, \overrightarrow{C_3}, D_3 \notin S_{1,2}$. (4) $\Rightarrow$ (3) By Lemma 5.2.9 and Lemma 5.5.24. (3) $\Rightarrow$ (2) Let $G' = (V, A')$ be a subdigraph of $G = (V, A)$ which is a transitive tournament. By Theorem 5.5.17 we know that $G' \in S_{1,1}$ and thus there is some sequence $q' = [v_1, \dots, v_n]$ such that $g(\{q'\}) = G'$. If $A' = A$ we know that $G \in S_{1,1} \subseteq S_{1,\ell}$ for every $\ell \geq 2$. So we can assume that $A' \subsetneq A$. Obviously for every arc $(v_i, v_j) \in A - A'$ there are two positions $j < i$ in $q' = [v_1, \dots, v_j, \dots, v_i, \dots, v_n]$. In order to define a subdigraph of G which contains all arcs of G' and arc (v_i, v_j) we can insert (cf. Section 5.5.1 for the definition of inserting an item) an additional item for type v_i on position $k \leq j$, or an additional item for type v_j on position $k > i$, or first an additional item for type v_j and then an additional item for type v_i on a position k, $j < k \leq i$, into q' without creating an arc which is not in A. This is possible if and only if there is some position k, $j \leq k \leq i$, in $q' = [v_1, \dots, v_j, \dots, v_{m'}, \dots, v_k, \dots, v_{m''}, \dots, v_i, \dots, v_n]$ such that for every m', $j < m' \leq k$, it holds $(v_{m'}, v_j) \in A$ and for every m'', $k \leq m'' < i$, it holds $(v_i, v_{m''}) \in A$.

If it is possible to insert all arcs of $A - A'$ by adding a set of additional items into sequence q' resulting in a sequence q such that $G = g(q)$, then it obviously holds $G \in S_{1,\ell}$ for some $\ell \geq 2$. Next we show a condition using the new items of every single arc of $A - A'$ independently from each other.

Claim 5.5.26. If for every arc $(v_i, v_j) \in A - A'$ there is a position k, $j < k \leq i$ such that first inserting an additional item for type v_j and then an additional item for type v_i at position k into q' defines a subdigraph of G which contains all arcs of G' and arc (v_i, v_j), then $G \in S_{1,\ell}$ for some $\ell \geq 2$.

Assume that $G \notin S_{1,\ell}$ for every $\ell \geq 2$. By the Claim there is some arc $(v_i, v_j) \in A - A'$ such that for every position k, $j < k \leq i$ inserting an additional item for type v_i and an additional item for type v_j at position k defines an arc which is not in A. That is, for every position k, $j < k \leq i$, in q' there exists some m', $j < m' \leq k$, such that it holds $(v_{m'}, v_j) \notin A$ *or* there exists some m'', $k \leq m'' < i$, such that it holds $(v_i, v_{m''}) \notin A$. By the transitivity of co-G it follows that there is one position k, $j < k \leq i$, in q' such that there exists some m', $j < m' \leq k$, such that it holds $(v_{m'}, v_j) \notin A$ *and* there exists some m'', $k \leq m'' < i$, such that it holds $(v_i, v_{m''}) \notin A$.

If co-$G = (V, A^c)$ is the complement digraph of G we know that

$$(v_{m'}, v_j) \in A^c \text{ and } (v_i, v_{m''}) \in A^c. \tag{5.8}$$

Since $m' \leq m''$ we know that $(v_{m'}, v_{m''}) \in A$. We also know that $(v_{m''}, v_{m'}) \in A$, since otherwise $(v_{m''}, v_{m'}) \in A^c$, property (5.8), and the transitivity of co-G would imply that $(v_i, v_j) \in A^c$ which is not possible. Thus we know that

$$(v_{m'}, v_{m''}) \notin A^c \text{ and } (v_{m''}, v_{m'}) \notin A^c. \tag{5.9}$$

Further the arcs $(v_j, v_{m'}), (v_j, v_{m''}), (v_{m'}, v_i), (v_{m''}, v_i)$ belong to $A' \subseteq A$ and thus

$$(v_j, v_{m'}) \notin A^c, (v_j, v_{m''}) \notin A^c, (v_{m'}, v_i) \notin A^c \text{ and } (v_{m''}, v_i) \notin A^c. \tag{5.10}$$

If $(v_i, v_{m'}) \in A^c$ or $(v_{m''}, v_j) \in A^c$ then (5.8) and the transitivity of co-G would imply that $(v_i, v_j) \in A^c$, thus we know

$$(v_i, v_{m'}) \notin A^c \text{ and } (v_{m''}, v_j) \notin A^c. \tag{5.11}$$

Properties (5.8)-(5.11) imply that $(\{v_i, v_j, v_{m'}, v_{m''}\}, \{(v_i, v_{m''}), (v_{m'}, v_j)\})$ induces a $2\overrightarrow{P_2}$ in co-G, which implies that G contains a co-$(2\overrightarrow{P_2})$. □

Corollary 5.5.27. *Every digraph in $S_{k,\ell}$ can be obtained by the union of at most k many $\{$co-$(2\overrightarrow{P_2}), 2\overleftrightarrow{K_1}, \overrightarrow{C_3}, D_3\}$-free digraphs.*

Proposition 5.5.28. *Let $G = (V, A) \in S_{1,2}$, then a sequence q, such that $G = g(\{q\})$ can be found in time $\mathcal{O}(|V| + |A|)$.*

Proof. Let $G = (V, A) \in S_{1,2}$ and $q = []$. We perform the following steps until $G = (\emptyset, \emptyset)$.

- Choose $v \in V$ such that $(v, u) \in A$ for all $u \in V - \{v\}$ and append v to q.
- Remove all arcs (v, u) from A.
- If indegree(v) = outdegree$(v) = 0$, remove v from V.
- If there are vertices u such that indegree(u) = outdegree$(u) = 0$, remove u from V and append u to q.

In order to perform the algorithm there has to be an ordering $v_1, \dots, v_n$ of V such that for $1 \leq i < n$ vertex v_i has maximum possible outdegree in subdigraph obtained by removing the outgoing arcs of $v_1, \dots, v_{i-1}$ and thereby created isolated vertices from G. Since $G \in S_{1,2}$ there is a sequence q' such that $G = g(\{q'\})$. The order in which the types corresponding to the vertices of V appear in subsequence $F(q')$, defined in the proof of Theorem 5.5.25, ensures the existence of such an ordering.

Finally it holds $G = g(\{q\})$ by the definition of sequence digraphs and since every vertex which has only outgoing or only incoming arcs will be inserted once into q and every vertex which has outgoing and incoming arcs will be inserted at most twice into q this sequence fulfils the properties stated in the theorem. □

5.5.4 Directed Path-width of Sequence Digraphs

Determining whether the (undirected) path-width of some given (undirected) planar graph with maximum vertex degree 3 is at most some given value w is NP-complete [MS88]. Since for complete bioriented digraphs the directed path-width (d-pw) is equal to the (undirected) path-width (pw) of the underlying undirected graph it follows that determining whether the directed path-width of some given digraph with maximum semi-degree $\Delta^0(G) = \max\{\Delta^-(D), \Delta^+(D)\} \leq 3$ is at most some given value w is NP-complete, which will be useful to show Proposition 5.5.30.

Hardness of Directed Path-width on Sequence Digraphs

Next we give some conditions on the sequences in Q such that for the corresponding digraph $g(Q)$ computing its directed path-width is NP-hard.

Proposition 5.5.29. *Given a set Q on k sequences such that $n_i = 2$ for $1 \leq i \leq k$ and an integer p, then the problem of deciding whether d-pw$(g(Q)) \leq p$ is NP-complete.*

Proof. The stated problem is in NP. To show the NP-hardness by a reduction from the directed path-width problem we transform instance (G, p) in linear time into instance $(q(G), p)$ for the stated problem. The correctness follows by Observation 5.5.6. □

Proposition 5.5.30. *Given a set Q with $d_Q = 3$ or $c_Q = 5$ and an integer p, then the problem of deciding whether d-pw$(g(Q)) \leq p$ is NP-complete.*

Proof. To show the NP-hardness by a reduction from the directed path-width problem for digraphs G with $\max\{\Delta^-(G), \Delta^+(G)\} \leq 3$, we transform instance (G, p) in linear time into instance $(q(G), p)$ for the stated problem. The correctness follows by Lemma 5.5.9. □

Polynomial Cases of Directed Path-width on Sequence Digraphs

We consider the directed path-width of sequence digraphs for $k = 1$ or $\ell = 1$.

Proposition 5.5.31. *Let $G \in S_{k,1}$, then d-pw$(G) = 0$.*

Proof. By Proposition 5.5.19 every digraph in $S_{k,1}$ is the disjoint union of k digraphs in $S_{1,1}$. By Theorem 5.5.17 every digraph in $S_{1,1}$ is acyclic and thus has directed path-width 0. □

For digraphs in $S_{1,2}$ the directed path-width can be arbitrary large, since this class includes all bidirectional complete digraphs. We can compute this value as follows. Let $Q = \{q\}$. For type $t \in \mathit{types}(q)$ let $I_t = [\mathit{first}(q,t), \mathit{last}(q,t)]$ be the interval representing t, and let $I_q = \{I_t \mid t \in \mathit{types}(q)\}$ be the set of all intervals for sequence q. Let $I(q) = (V,E)$ be the interval graph where $V = \mathit{types}(q)$ and $E = \{\{u,v\} \mid u \neq v,\ I_u \cap I_v \neq \emptyset, I_u, I_v \in I_q\}$.

Proposition 5.5.32. *Let $G \in S_{1,2}$ defined by a set $Q = \{q_1\}$ of one sequence, then d-pw$(G) = \omega(I(q)) - 1 = pw(I(q))$.*

Proof. We obtain d-pw$(G) \leq \omega(I(q)) - 1$ by an obvious directed path-decomposition along $I(q)$. Further for every integer r the set $I(r) = \{I_t \mid r \in I_t\}$ defines a complete subgraph $K_{|I(r)|}$ in $I(q)$ and also a bidirectional complete subdigraph $\overleftrightarrow{K_{|I(r)|}}$ in G. Thus it holds d-pw$(G) \geq \omega(I(q)) - 1$. The second equality holds since the (undirected) path-width of an interval graph is equal to the size of a maximum clique [Bod98]. □

Sets Q where $d_Q = 1$ can be handled in polynomial time.

Proposition 5.5.33. *Given a set Q with $d_Q = 1$ and an integer p, then the problem of deciding whether d-pw$(g(Q)) \leq p$ can be solved in time $\mathcal{O}(|\mathit{types}(Q)|^2 + n)$.*

Proof. Let $Q = \{q_1, \ldots, q_k\}$. If $d_Q = 1$ the vertex sets $V_i = \mathit{types}(q_i)$ are disjoint. That is, $g(Q)$ is the disjoint union of digraphs in $S_{1,2}$ for which the directed path-width can be computed in time $\mathcal{O}(\sum_{i=1}^{k} |\mathit{types}(\{q_i\})|^2 + n_i) \subseteq \mathcal{O}(|\mathit{types}(Q)|^2 + n)$ by Proposition 5.5.32. □

An XP-Algorithm for Directed Path-width

We next give an XP-algorithm for directed path-width w.r.t. the parameter k, which implies that for every constant k for a given set Q on at most k sequences the value d-pw$(g(Q))$ can be computed in polynomial time. The main idea is to discover an optimal directed path-decomposition by scanning the k sequences left-to-right and keeping in a state the numbers of scanned items of every sequence and a certain number of active types.

Let $Q = \{q_1, \ldots, q_k\}$ be a set of k sequences. Every k-tuple $(i_1, \ldots, i_k)$ where $0 \leq i_j \leq n_j$ for $1 \leq j \leq k$ is a *state* of Q. State $(0, 0, \ldots, 0)$ is the *initial state* and $(n_1, \ldots, n_k)$ is the *final state.* The *state digraph $s(Q)$* for a set Q has a vertex for each possible state. There is an arc from vertex u labeled by $(u_1, \ldots, u_k)$ to vertex v labeled by $(v_1, \ldots, v_k)$ if and only if $u_i = v_i - 1$ for exactly one element of the vector and for all other elements of the vector $u_j = v_j$. Let $(i_1, \ldots, i_k)$ be a state of Q. We define $L(i_1, \ldots, i_k)$ to be the set of all items on the positions $1, \ldots, i_j$ for $1 \leq j \leq k$ and $R(i_1, \ldots, i_k)$ is the set of all items on the remaining positions $i_j + 1, \ldots, n_j$ for

$1 \leq j \leq k$. Further let $M(i_1, \ldots, i_k)$ be the set of all items on the positions i_j for $1 \leq j \leq k$ such that there is exactly one type of these items in Q. Obviously, for every state $(i_1, \ldots, i_k)$ it holds that $L(i_1, \ldots, i_k) \cup R(i_1, \ldots, i_k)$ leads to a disjoint partition of the items in Q and $M(i_1, \ldots, i_k) \subseteq L(i_1, \ldots, i_k)$.

Further each vertex v of the state digraph is labeled by the value $f(v)$. This value is the number of types t such that either there is at least one item of type t in $L(v)$ and at least one item of type t in $R(v)$ or there is one item of type t in $M(v)$. Formally we define $\mathit{active}(v) = \{t \in \mathit{types}(Q) \mid b \in L(v), t(b) = t, b' \in R(v), t(b') = t\} \cup \{t \in \mathit{types}(Q) \mid b \in M(v), t(b) = t\}$ and $f(v) = |\mathit{active}(v)|$. Obviously for the initial state v it holds $|\mathit{active}(v)| = 0$. Since the state digraph $s(Q)$ is a directed acyclic graph we can compute all values $|\mathit{active}(v)|$ using a topological ordering *topol* of the vertices. Every arc (u, v) in $s(Q)$ represents one item $b_{i,j}$ if item $b_{i,j-1} \notin M(v)$ and two items $b_{i,j}$ and $b_{i,j-1}$ if item $b_{i,j-1} \in M(v)$ of some types $t(b_{i,j}) = t$ and $t(b_{i,j-1}) = t'$ from some sequence q_j, thus

$$\begin{aligned} &|\mathit{active}((i_1, \ldots, i_{j-1}, i_j + 1, i_{j+1}, \ldots, i_k))| \\ &\quad = \ |\mathit{active}((i_1, \ldots, i_{j-1}, i_j, i_{j+1}, \ldots, i_k))| + c_j \end{aligned}$$

where

$$c_j = \begin{cases} 1, & \text{if } \mathit{first}(q_j, t) = i_j + 1 \text{ and } \mathit{first}(q_\ell, t) > i_\ell \ \ \forall\, \ell \neq j \ \text{ and} \\ & \text{not}(\mathit{first}(q_j, t') = \mathit{last}(q_j, t') = i_j \text{ and } \mathit{last}(q_\ell, t') = 0 \ \ \forall\, \ell \neq j) \\ 0, & \text{if } \mathit{first}(q_j, t) = i_j + 1 \text{ and } \mathit{first}(q_\ell, t) > i_\ell \ \ \forall\, \ell \neq j \ \text{ and} \\ & \mathit{first}(q_j, t') = \mathit{last}(q_j, t') = i_j \text{ and } \mathit{last}(q_\ell, t') = 0 \ \ \forall\, \ell \neq j \\ -1, & \text{if } \mathit{last}(q_j, t) = i_j + 1 \text{ and } \mathit{last}(q_\ell, t) \leq i_\ell \ \ \forall\, \ell \neq j \ \text{ and} \\ & \text{not}(\mathit{first}(q_j, t') = \mathit{last}(q_j, t') = i_j \text{ and } \mathit{last}(q_\ell, t') = 0 \ \ \forall\, \ell \neq j) \\ -2, & \text{if } \mathit{last}(q_j, t) = i_j + 1 \text{ and } \mathit{last}(q_\ell, t) \leq i_\ell \ \ \forall\, \ell \neq j \ \text{ and} \\ & \mathit{first}(q_j, t') = \mathit{last}(q_j, t') = i_j \text{ and } \mathit{last}(q_\ell, t') = 0 \ \ \forall\, \ell \neq j \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the calculation of value $|\mathit{active}(i_1, \ldots, i_k)|$ for the vertex labeled $(i_1, \ldots, i_k)$ depends only on already calculated values, which is necessary in order to use dynamic programming.[5]

Let $\mathcal{P}(Q)$ the set of all paths from the initial state to the final state in $s(Q)$. Every $P \in \mathcal{P}(Q)$ has $r = 1 + \sum_{i=1}^{k} n_i$ vertices, i.e. $P = (v_0, \ldots, v_r)$. First we show that every path in $\mathcal{P}(Q)$ leads to a directed path-decomposition for $g(Q)$.

Lemma 5.5.34. *Let Q be a set of k sequences and $(v_0, \ldots, v_r) \in \mathcal{P}(Q)$. Then $(\mathit{active}(v_1), \ldots, \mathit{active}(v_{r-1}))$ is a directed path-decomposition for $g(Q)$.*

Lemma 5.5.34 leads to an upper bound on the directed path-width of $g(Q)$ using the state graph. The reverse direction is more involved and considered next.

[5] For sets Q such that the number of items for which there is no further item of the same type in Q is small, we suggest to modify Q by inserting a dummy item of the same type at the position after such items. This does not change the sequence digraph and allows to make a case distinct within three instead of five cases when defining c_j. But this modification increases the size of the sequence digraph.

Lemma 5.5.35. *Let Q be a set of k sequences. If there is a directed path-decomposition of width $p-1$ for $g(Q)$, then there is a path $(v_0,\dots,v_r) \in \mathcal{P}(Q)$ such that for every $1 \leq i \leq r$ it holds $|active(v_i)| \leq p$.*

By Lemma 5.5.34 and Lemma 5.5.35 we obtain the following result.

Corollary 5.5.36. *Given a set Q of k sequences, then*

$$d\text{-}pw(g(Q)) = \min_{(v_0,\dots,v_r)\in\mathcal{P}(Q)} \max_{1\leq i\leq r-1} |active(v_i)| - 1.$$

In order to apply Corollary 5.5.36 we consider some general digraph problem. Let $G=(V,A,f)$ be a directed acyclic vertex-labeled graph. Function $f: V \to \mathbb{Z}$ assigns to every vertex $v \in V$ a value $f(v)$. Let $s \in V$ and $t \in V$ be two vertices. For some vertex $v \in V$ and some path $P=(v_1,\dots,v_\ell)$ with $v_1 = s$, $v_\ell = v$ and $(v_i,v_{i+1}) \in A$ we define $val_P(v) := \max_{u\in P}(f(u))$. Let $\mathcal{P}_s(v)$ denote the set of all paths from vertex s to vertex v. We define $val(v) := \min_{P\in\mathcal{P}_s(v)}(val_P(v))$. Then it holds:

$$val(v) = \max\{f(v), \min_{u\in N^-(v)}(val(u))\}.$$

By dynamic programming it is possible to compute all the values of $val(v)$, $v \in V$, in time $\mathcal{O}(|V|+|A|)$. This is possible, since G is acyclic.

Theorem 5.5.37. *Given a set Q, such that $g(Q) \in S_{k,\ell}$ for some $\ell \geq 1$, then the directed path-width of $g(Q)$ and also a directed path-decomposition can be computed in time $\mathcal{O}(k \cdot (1+\max_{1\leq i\leq k} n_i)^k)$.*

Proof. Let Q be a set, such that $g(Q) \in S_{k,\ell}$. The state digraph $s(Q)$ has at most $(1+\max_{1\leq i\leq k} n_i)^k$ vertices and can be found in time $\mathcal{O}(k\cdot(1+\max_{1\leq i\leq k} n_i)^k)$ from Q. By Corollary 5.5.36 the directed path-width of $g(Q)$ can be computed by considering all paths from the initial state to the final state in $s(Q)$. This can be done by any algorithm for the above general digraph problem on $s(Q) = (V,A)$ using $f(v) = |active(v)|$, $v \in V$, s as the initial state, and t as the final state. Since every vertex of the state digraph has at most k outgoing arcs we have $\mathcal{O}(|V|+|A|) \subseteq \mathcal{O}(k \cdot (1+\max_{1\leq i\leq k} n_i)^k)$. Thus we can compute an optimal path in $s(Q)$ in time $\mathcal{O}(k \cdot (1+\max_{1\leq i\leq k} n_i)^k)$. □

5.5.5 Conclusions

In this section, we have considered digraphs which can be defined by a set of k sequences. We have shown an XP-algorithm for directed path-width w.r.t. number of sequences k needed to define the input graph. For special digraphs our solution improves known solution w.r.t. the standard parameter as shown in Table 5.2. This implies that for each constant k, it is decidable in polynomial time whether for a given set Q on at most k sequences the digraph $g(Q)$ has directed path-width at most w. If we know that some digraph can be defined by one sequence, we can find this in linear time (Proposition 5.5.28). This implies that for each constant k, it is decidable in polynomial time whether for a digraph G, which is given by the union of at most k many semicomplete $\{\text{co-}(2\overrightarrow{P_2}), \overrightarrow{C_3}, D_3\}$-free digraphs, digraph G has directed path-width at most w.

There are several interesting open questions. **(a)** Is there is an FPT-algorithm for the directed path-width problem w.r.t. parameter k? **(b)** Does the hardness of Proposition 5.5.30 also hold for $c_Q \in \{2, 3, 4\}$ and for $d_Q = 2$? **(c)** By Theorem 5.5.17, Proposition 5.5.19 and Theorem 5.5.25 one can decide in polynomial time whether a given digraph belongs to the class $S_{k,\ell}$ for $\ell = 1$, for $k = 1$, or both. It remains to consider this problem for the classes $S_{k,\ell}$ for $k \geq 2$ and $2 \leq \ell \leq 2k$. **(d)** Can we find for a given digraph G a set Q with a smallest number of sequences such that $g(Q) = G$ in polynomial time?

5.6 Semicomplete Graphs

The map of relations between the different directed width measures in general still has some blank spots. In this section we fill in many of these open relations for the restricted class of semicomplete digraphs. To do this we show the equivalence between the parameterss directed path-width, directed tree-width, DAG-width, and Kelly-width. Moreover, we show that directed (linear) clique-width is upper bounded in a function of directed tree-width on semicomplete digraphs. This allows for a quadratic approximation of many of these parameters by using the known FPT-approximation algorithm for directed tree-width. Moreover, in general the algorithmic use of many of these parameters is fairly restricted. On semicomplete digraphs our results allow to combine the nice computability of directed tree-width with the algorithmic power of directed clique-width.

5.6.1 Linear Width parameters on semicomplete graphs

For semicomplete digraphs the directed path-width can be used to give an upper bound on the directed clique-width, which has been shown in [FP13a]. The main idea of the proof is to define a directed clique-width expression along a nice path-decomposition.[6] Since the proof only uses directed linear clique-width operations we can state the next theorem.

This subsection is taken from [GR19a].

Lemma 5.6.1 ([FP13a])**.** *For every semicomplete digraph S, we have*

$$d\text{-}lcw(S) \leq d\text{-}pw(S) + 2.$$

[6]Please note that in [FP13a] a different notation for directed path-width was used. In Definition 3.2.1(dpw-2) the arcs are directed from bags X_i to X_j for $i \leq j$. The authors of [FP13a] take arcs from bags X_i to X_j for $i \geq j$ into account. Since an optimal directed path-decomposition $(X_1, \ldots, X_r)$ w.r.t. Definition 3.2.1 leads to an optimal directed path-decomposition $(X_r, \ldots, X_1)$ w.r.t. the definition of [FP13a], and vice versa, both definitions lead to the same value for the directed path-width.

Lemmas 5.6.1, 3.5.7, 3.5.8, and 3.5.9 imply the following bounds.

Corollary 5.6.2. *For every semicomplete digraph S, we have*

$$\textit{d-lnlcw}(S) \leq \textit{d-pw}(S) + 2,$$

$$\textit{d-nw}(S) \leq \textit{d-pw}(S) + 2, \textit{ and}$$

$$\textit{d-lrw}(S) \leq \textit{d-pw}(S) + 2.$$

Theorem 5.6.3. *For every class of semicomplete digraphs $\mathcal{G}$ such that for all $G \in \mathcal{G}$ the value $\min(\Delta^-(G), \Delta^+(G))$ is bounded any two parameters in $\{$d-cutw, d-pw, d-nw, d-lnlcw, d-lcw, d-lrw$\}$ are equivalent.*

Using the results of Theorem 3.5.1(a), [FOT10, Section 8], and Theorem 3.5.1(f), respectively, there is some polynomial p such that for every digraph G, we have

$$\text{d-pw}(G) \leq \text{pw}(\textit{und}(G)) \leq p(\Delta(\textit{und}(G)), \text{lrw}(\textit{und}(G))) \leq p(\Delta(G), \text{d-lrw}(G)).$$

This allows us to strengthen the result of Theorem 5.6.3 as follows.

Theorem 5.6.4. *For every class of semicomplete digraphs $\mathcal{G}$ such that for all $G \in \mathcal{G}$ the value $\Delta(G)$ is bounded any two parameters in $\{$d-cutw, d-pw, d-nw, d-lnlcw, d-lcw, d-lrw$\}$ are polynomially equivalent.*

Except for directed linear rank-width we even have shown linear equivalence.

Theorem 5.6.5. *For every class of semicomplete digraphs $\mathcal{G}$ such that for all $G \in \mathcal{G}$ the value $\min(\Delta^-(G), \Delta^+(G))$ is bounded any two parameters in $\{$d-cutw, d-pw, d-nw, d-lnlcw, d-lcw$\}$ are linearly equivalent.*

By Lemmas 3.5.19 and 5.6.1 the restriction to semicomplete digraphs[7] leads to the same relation between path-width and linear clique-width as for undirected graphs (see [Gur06b]).

5.6.2 Non-Linear Width parameters on semicomplete digraphs

Comparing non-linear width parameters to each other and to the linear parameter of directed path-width on semicomplete digraphs also leads to algorithmically interesting results.

Especially the relation between directed clique-width and the tree-based parameters of directed tree-width, DAG-width and Kelly-width is worth investigating, as they have very different algorithmic properties. Huge parts of this section are taken from [GKRW21].

[7] When considering the directed path-width of almost semicomplete digraphs in [KKT15] the class of semicomplete digraphs was suggested to be "a promising stage for pursuing digraph analogues of the splendid outcomes, direct and indirect, from the Graph Minors project".

g \ f	d-pw	d-tw	dagw	kw	d-lcw	d-cw
d-pw	k	$4k^2 + 15k + 10$	$k - 1$	$4k^2 + 7k$	∞	∞
d-tw	k	k	$k - 1$	$6k - 2$	∞	∞
dagw	$k + 1$	$4k^2 + 15k + 11$	k	k^2	∞	∞
kw	$k + 1$	$4k^2 + 15k + 11$	k	k	∞	∞
d-lcw	$k + 2$	$4k^2 + 15k + 12$	$k + 1$	$k^2 + 2$	k	∞
d-cw	$k + 2$	$4k^2 + 15k + 12$	$k + 1$	$k^2 + 2$	k	k

Table 5.4: Relations between digraph parameter on **semicomplete digraphs**. The parameter of the left column is bounded by the respective parameter of the top row by the specified function where k is the corresponding width. We use '∞' if the relation is unbounded, that is if $h_{f,g}$ does not exist.

Directed path-width, directed tree-width, DAG-width and Kelly-width correspond to different variants of so-called cops and robber games. Width parameters corresponding to variants of the cops & robber game have the inherent advantage of coming with an XP-time (approximation) algorithm for finding a decomposition of (almost) optimal width. They also tend to correlate with structural properties and thus, as exemplified by tree-width, make for great tools for structure theory. However, strong evidence exists that for digraphs no such parameter can, in addition to these advantages, replicate the algorithmic power of tree-width in undirected graphs [GHK+16].

An algorithmically stronger parameter is directed clique-width (d-cw) [CMR00] by Courcelle et al., a width measure which is, in essence, defined for relational structures and whose algorithmic properties do not distinguish between graphs and digraphs. Hence directed clique-width does not suffer from the sudden increase in complexity when transitioning from graphs to digraphs and the existence of a powerful algorithmic meta theorem is preserved: Every problem expressible in MSO_1 logic is fixed parameter tractable with respect to the parameter directed clique-width [CMR00]. Still, directed clique-width has its drawbacks, as there is no known direct way to compute a bounded width expression. The current method to obtain such an expression is by approximating birank-width which leads to an exponential approximation of directed clique-width [OS06]. Unfortunately, directed clique-width is in general incomparable to the previously mentioned tree-width inspired parameters. So in general the nice computability properties of the decompositions relating to variants of the cops and robber game cannot be used to obtain bounded width expression for directed clique-width.

Semicomplete digraphs are a superclass of tournaments which received significant attention in the past [CS11, KS15]. In this section we show that on semicomplete digraphs, all of the path-width and tree-width inspired parameters are equivalent.

Indeed, all of these equivalences are realized by relatively tame functions obtained without complicated proofs.

As by [FP19] for a semicomplete digraph G it holds that d-cw(G) is at most d-pw(G) + 2, we conclude that all above mentioned parameters are upper bounds to directed clique-width. This result is even extendable to directed linear clique-width (d-lcw). More precisely we show in this section that, for any choice of functions $f, g \in \{\text{d-pw}, \text{d-tw}, \text{dagw}, \text{kw}, \text{d-lcw}, \text{d-cw}\}$, there exists a function $h_{f,g}$ such that, if G is a semicomplete digraph with $f(G) \leq k$ then $g(G) \leq h_{f,g}(k)$ where the functions $h_{f,g}$ are presented in Table 5.4 if they exist.

Theorem 5.6.6. *Let G be a semicomplete digraph and $f, g \in \{\textit{d-pw}, \textit{d-tw}, \textit{dagw}, \textit{kw}, \textit{d-lcw}, \textit{d-cw}\}$. If $f(G) \leq k$, then $g(G) \leq h_{f,g}(k)$ where $h_{f,g} \colon \mathbb{N} \to \mathbb{N}$ is given by Table 5.4.*

Combining these results with the above mentioned theorem of Courcelle et al. on bounded directed clique-width [CMR00] and the FPT-algorithm for approximating directed tree-width within a linear factor by Campos et al. [CLMS19], this leads to the following result:

Theorem 5.6.7. *Every problem expressible in MSO_1 logic is fixed parameter tractable on semicomplete digraphs with respect to the parameter directed tree-width.*

5.6.3 DAG-width and directed path-width on semicomplete digraphs

As a first step towards Theorem 5.6.6 we show that DAG-width plus 1 and directed path-width are equal on the class of semicomplete digraphs, which leads also to the fact that computing DAG-width of a semicomplete digraph is in NP.

This later fact might be of independent interest since DAG-width is PSPACE-complete in general [AKK$^+$15], but, it is one of only few known parameters from the tree-width inspired family which allows for an efficient solving of parity games [BDHK06].

As a tool we need a normalized version of DAG-decompositions.

Definition 5.6.8 (Nice DAG-decomposition)**.** A DAG-decomposition $(D, \mathcal{X})$ of a digraph G is *nice*, if the following properties are fulfilled.

1. D has exactly one source r.
2. Every vertex in D has at most two successors.
3. If vertex d has two successors d' and d'', then $X_d = X_{d'} = X_{d''}$.
4. If vertex d has one successors d', then $|(X_d \setminus X_{d'}) \cup (X_{d'} \setminus X_d)| = 1$.

Berwanger et al. [BDHK06] showed that if digraph G has a DAG-decomposition of width k, it also has a nice DAG-decomposition of width k. Moreover, since deleting transitive edges from D does neither destroy any of the properties of a DAG-decomposition, nor increase the width of the DAG-decomposition, we get the following property.

Lemma 5.6.9. *If digraph G has a DAG-decomposition of width k, it also has a nice DAG-decomposition $(D, \mathcal{X})$ of width k such that D has no transitive edges.*

Proposition 5.6.10. *For every semicomplete digraph G it holds that d-pw$(G) \leq$ dagw$(G) - 1$.*

Proof. Let G be a semicomplete digraph and let $(D, \mathcal{X})$ be a nice DAG-decomposition for G of width k with digraph D, vertex set V_D and $\mathcal{X} = \{X_u \mid u \in V_D\}$. By Lemma 5.6.9 we can assume that D has exactly one source, every vertex in D has at most two successors and no transitive edges. We show that in case D is not a path, we can convert it into a path without increasing the width. Assume D is not a path. For any vertex r let V_{D_r} is the set of vertices of D which are reachable from r. Let D_t be the maximal subdigraph of D with unique source t. Consider vertex $q \in V_D$ with two successors s and t. We differentiate three cases: All vertices from G which are in bags of D_s are also in the bags of D_t (Case 1.a), the opposite inclusion (Case 1.b) or, at last none of these inclusions (Case 2) occur.

Case 1.a: $(\bigcup_{u \in V_{D_s}} X_u) \cup X_q \subseteq (\bigcup_{u \in V_{D_t}} X_u) \cup X_q$.

In order to define a new DAG-decomposition $(D', \mathcal{X}')$ for G, we simply remove all vertices $V_{D_s} \backslash V_{D_t}$ from D and forget all bags associated with removed vertices. We now show that $(D', \mathcal{X}')$ is a DAG-decomposition for G by checking the conditions of the definition.

- (dagw-1) Is satisfied since

$$\bigcup_{u \in V_{D'}} X_u = \bigcup_{u \in V_D \backslash V_{D_s}} X_u \cup \bigcup_{u \in V_{D_t}} X_u \overset{(*)}{\supseteq} \bigcup_{u \in V_D \backslash V_{D_s}} X_u \cup \bigcup_{u \in V_{D_s}} X_u = \bigcup_{u \in V_D} X_u = V_G$$

  The inclusion in $(*)$ holds by assumption of case 1a) since $q \in V_D \backslash V_{D_s}$.

- (dagw-2) is still satisfied since for every $a, b, c \in V_{D'}$ it holds that if $a \preccurlyeq_{D'} b \preccurlyeq_{D'} c$ then

$$X'_a \cap X'_b = X_a \cap X_c \subseteq X_b = X'_b$$

- (dagw-3) Let $(a, b) \in E_{D'}$, then it follows that $(a, b) \in E_D$. Therefore, it must hold that $X_a \cap X_b$ guards $X_{\succcurlyeq b} \backslash X_a$. It holds that $X'_a = X_a$ and $X'_b = X_b$. Further, $X'_{\succcurlyeq b}$ is the union of all bags of vertices that we can reach from vertex b in D', such that $X'_{\succcurlyeq b} = \bigcup_{b \preccurlyeq_{D'} u} X_u$.

  (i) If $b \preccurlyeq_{D'} t$, then:

$$X'_{\succcurlyeq b} = \bigcup_{b \preccurlyeq_{D'} u \preccurlyeq_{D'} t} X_u \cup \bigcup_{t \preccurlyeq_{D'} u} X'_u = \bigcup_{b \preccurlyeq_D u \preccurlyeq_D t} X_u \cup \bigcup_{t \preccurlyeq_D u} X_u$$

$$\text{(since } X_q \subseteq \bigcup_{b \preccurlyeq_D u \preccurlyeq_D t} X_u)$$

$$= \bigcup_{b \preccurlyeq_D u \preccurlyeq_D t} X_u \cup \bigcup_{t \preccurlyeq_D u} X_u \cup \bigcup_{s \preccurlyeq_D u} X_u = \bigcup_{b \preccurlyeq_D u} X_u = X_{\succcurlyeq b}$$

(ii) Else $t \prec_{D'} b$, then: Since every successor of b in D is also in D' it holds that

$$X'_{\succcurlyeq b} = \bigcup_{b \preccurlyeq_{D'} u} X_u = \bigcup_{b \preccurlyeq_D u} X_u = X_{\succcurlyeq b}$$

This leads to $X'_a \cap X'_b = X_a \cap X_b$ guards $X'_{\succcurlyeq b} \backslash X'_a = X_{\succcurlyeq b} \backslash X_a$.

Thus, all requirements of a DAG-decomposition are met by $(D', \mathcal{X}')$.

Case 1.b: $(\bigcup_{u \in V_{D_t}} X_u) \cup X_q \subseteq (\bigcup_{u \in V_{D_s}} X_u) \cup X_q$ can be handled analogously to case 1.a.

Case 2: $(\bigcup_{u \in V_{D_s}} X_u) \cup X_q \nsubseteq (\bigcup_{u \in V_{D_t}} X_u) \cup X_q$ and $(\bigcup_{u \in V_{D_t}} X_u) \cup X_q \nsubseteq (\bigcup_{u \in V_{D_s}} X_u) \cup X_q$. More informally, this means that there exist vertices from G that are only represented in bags of D_s but not in bags of D_t. We show now, that this case cannot occur. There are x, y such that

$$x \in X_q \cup \bigcup_{u \in V_{D \geq s}} X_u, x \notin X_q \cup \bigcup_{u \in V_{D \geq t}} X_u \tag{5.12}$$

$$y \notin X_q \cup \bigcup_{u \in V_{D \geq s}} X_u, y \in X_q \cup \bigcup_{u \in V_{D \geq t}} X_u \tag{5.13}$$

Since G is semicomplete, there is an arc between x and y in G. W.l.o.g. let $(x, y) \in E_G$. By the connectivity property given by (dagw-2) it holds that $x, y \notin \bigcup_{u \preccurlyeq_D q} X_u$, since $x, y \notin X_q$. Let $w \in V_D, x \in X_w, x \notin X_u$ and $u \preccurlyeq_D w$. As equation (5.12) holds, this leads to $s \preccurlyeq_D w$. By (dagw-3) it further holds that $X_{w'} \cap X_w$ guards $X_{\succcurlyeq w} \backslash X_{w'}$ for a predecessor w' of w in D with $w' \neq s$. This means that for all $(z, z') \in E_G$ with $z \in X_{\succcurlyeq w} \backslash X_{w'}$ it holds that $z' \in (X_{\succcurlyeq w} \backslash X_{w'}) \cup (X_{w'} \cap X_w)$.

As assumed before, it holds that $(x, y) \in E_G$ with $x \in X_{\succcurlyeq w} \backslash X_{w'}$. By equation (5.13) it holds that $y \notin X_{w'} \cap X_w \Rightarrow y \in X_{\succcurlyeq w} \backslash X_{w'}$. By equation (5.13) it holds that $y \notin X_{w'} \Rightarrow y \in X_{\succcurlyeq w} = \bigcup_{w \preccurlyeq_D u} X_u$. But since $s \preccurlyeq_D w$ it holds that $\bigcup_{w \preccurlyeq_D u} X_u \subseteq \bigcup_{s \preccurlyeq_D u} X_u$. This contradicts that by equation (5.13) it holds that $y \notin \bigcup_{s \preccurlyeq_D u} X_u$. This leads to the conclusion that case 2 cannot occur.

Consequently, starting at the root, we can transform every DAG D of a DAG-decomposition of the semicomplete digraph G into a directed path. Since directed path-width is exactly the path variant of DAG-width, d-pw$(G) \leq$ dagw$(G) - 1$ holds. □

By Proposition 5.6.10 we can conclude that on semicomplete digraphs, DAG-width and path-width are equal.

Corollary 5.6.11. *For every semicomplete digraph G it holds that*

$$\textit{d-pw}(G) + 1 = \textit{dagw}(G)$$

5.6.4 Escaping pursuit in the jungle: directed path-width, directed tree-width and Kelly-width

Fradkin and Seymour [FS13] gave a description of semicomplete digraphs of bounded directed path-width. Indeed, they proved that every semicomplete digraph of huge directed path-width must contain a subdivision of a large bioriented clique [FS13]. While this result immediately implies that directed path-width acts, parametrically, as a lower bound for all tree-width inspired directed width measures discussed in this section, the proof uses a Ramsey argument and thus, for G to contain a subdivision of the complete biorientation of K_t, the directed path-width must be exponential in t. However, Fradkin and Seymour introduced another obstruction to small directed path-width on semicomplete digraphs which is similar to the idea of well linked sets. With a bit of more careful analysis we are able to obtain the quadratic bounds of Theorem 5.6.6.

Note that [FS13] could also be used for comparisons between directed path-width and DAG-width, but this would only lead to equivalence between those parameters, whereas we could prove equality (plus 1).

Two vertices u, v are *k-connected*, if there are at least k internally-disjoint paths from u to v and from v to u. For digraph $G = (V, E)$ a set $U \subseteq V$ is a *k-jungle* in G if $|U| = k$ and for all $u, v \in U$ it holds that u and v are k-connected.

For both, directed tree-width and Kelly-width, we show that the existence of a $k+1$-jungle is enough to ensure a winning strategy for the robber against k cops in the respective variants of of cops & robber game. Let us start with directed tree-width.

Proposition 5.6.12. *Let G be a semicomplete digraph. If d-pw$(G) \geq 4(k+1)^2 + 7(k+1)$ then d-tw$(G) \geq k$.*

Proof. Let us assume d-pw$(G) \geq 4(k+1)^2 + 7(k+1)$. Then, by the results from [FS13], we know that $G = (V, E)$ contains a $k+1$-jungle $J \subseteq V$. If we can show that the existence of J is enough to ensure that k-cops cannot catch the robber in the visible strong component cops and robber game on G, it follows from Proposition 3.3.39 that the directed tree-width of G must be at least k and thus the assertion follows. Hence what is left to do is describe a winning strategy for the robber against k cops on a $k+1$-jungle J. For the first position (C_0, r_0) we have $C_0 = \emptyset$ and the robber may select r_0 to be any vertex of J. Now suppose the game has been going on for i rounds and in each round the robber was able to select a vertex of J as her position. Let (C_{i-1}, r_{i-1}) be the current state of the game and let $C_i \subseteq V$ be the next position of the cops. In case $r_{i-1} \notin C_i$ there is nothing to do for the robber and she can stay where she is i.e. $r_i \coloneqq r_{i-1}$. So we may assume $r_{i-1} \in C_i$. In this case we know $|C_i \setminus \{r_{i-1}\}| \leq k-1$ and thus $|C_{i-1} \cap C_i| \leq k-1$. Hence there must exist a vertex $v \in J \setminus C_i$. As $r_{i-1} \neq v$ we know from J being a $k+1$-jungle that there exist $k+1$ pairwise internally disjoint paths from r_{i-1} to v and vice versa. As $|C_i| \leq k$ in $G - (C_{i-1} \cap C_i)$ at least one path from r_{i-1} to u and one from u to r_{i-1} must be left and thus both vertices belong to the same strong component of $G - (C_{i-1} \cap C_i)$. Thus v is reachable from r_{i-1} and we may set $r_i \coloneqq v$. As the robber was able to flee to another vertex of J our claim now follows by induction. □

From [AKK+15] and Corollary 5.6.11 we already know an upper bound on directed path-width in terms of Kelly-width, which is $\text{d-pw}(G) \leq 72\text{kw}(G)^2 + 1$. We can improve this bound following the same general idea as given above. Indeed, since in the strategy as described in the proof of Proposition 5.6.12 the robber only changed her position if she was threatened to be caught if she did not, the strategy above is already a strategy for a visible robber in the strong component game. Since the reachability searching game is a relaxation of the strong component game and the (in)visibility of the robber does not play a role in this strategy it is straight forward to see that using the same technique, an invisible and inert robber can also avoid being caught by k cops in the reachability searching game. From these arguments we obtain the following result.

Proposition 5.6.13. *Let G be a semicomplete digraph. If $\text{d-pw}(G) \geq 4(k+1)^2 + 7(k+1)$ then $kw(G) \geq k$.*

5.6.5 Directed (linear) clique-width and directed path-width on semicomplete digraphs

As already mentioned in the previous section in Corollary 5.6.2, in [FP19], the authors prove that on semicomplete digraphs, directed path-width can be used to give an upper bound for directed clique-width, which can be generalized to directed linear clique-width.

Note that the other direction, i.e. using directed (linear) clique-width as an upper bound of directed path-width, is not possible for semicomplete digraphs in general. That follows directly from the proof of Proposition 3.5.25, as the counterexample, a bioriented clique, is a semicomplete digraph.

Using the results from this and previous subsections, it is possible to improve the general results for the comparison of directed width parameters on semicomplete digraphs.

By using Proposition 5.6.10, 5.6.12, 5.6.13 and Corollary 5.6.2 we improve also other bounds between directed width parameters on semicomplete digraphs.

5.6.6 Conclusion

The landscape of directed width measures is a wild one. Started by the introduction of directed tree-width many different generalizations of undirected tree-width have been invented and received different amounts of attention. Some of these parameters were considered very little; possibly because of the results of [GHK+16], which essentially rule out any algorithmic application of these parameters beyond some specialized routing problems. So while the search for 'good' digraph width parameters inspired by tree-width does not seem very promising, one could turn to the logic based parameters instead. Here directed clique-width reigns supreme, but recently other attempts at finding interesting parameters such as a directed version of *maximum induced matching width* [JKT21] have been made.

In this section we have shown the equivalence of directed path-width, directed tree-width, Kelly-width and DAG-width on semicomplete digraphs. In particular this implies that each of these measures acts as an upper bound on directed clique-width and thus the algorithmic power of directed clique-width can now be accessed by any of the other parameters. Hence as a consequence of our results every digraph problem, which is describable in MSO_1 logic is fixed parameter tractable for these width measures for a given decomposition on semicomplete graphs.

Our result, that computing DAG-width is in NP on semicomplete digraphs while it is PSPACE-hard in general [AKR16], recalls the question if computing directed path-width and thus, DAG-width is NP-hard on semicomplete digraphs, though there are FPT algorithms to solve this problem [FP19].

6 Conclusions and Outlook

In this work, we give a discussion on the best known directed graph parameters. Especially we give comparisons and regard computability of these parameters, firstly in general and then on special directed graph classes.

While directed linear graph parameters are well comparable in general [GR19a], this is not true for directed non-linear width measures. Tables giving an overview about the general relations can be found in chapter 3.5.

But this changes on restricted graph classes.

The smallest considered class in this work are tree-like digraphs, especially directed pseudoforests and directed cactus forests [GR19b]. Those classes are very useful, as they are definable by forbidden directed graph minors and they further have strongly bounded directed path-width and directed tree-width. We show that directed cactus forests and its subclasses have d-tw at most 1 while directed pseudoforests and their subclasses even have d-pw at most 1. This is a remarkable result, as it has been a long considered matter to obtain all graphs of d-pw and d-tw at most 1.

Further, we extensively consider directed co-graphs [GR18, GKR19a, GKR21b] and show, that several of the most important graph parameters as d-pw, d-tw, dagw$-$1 and cr are not even equivalent, but exactly equal on this graph class. For kw, equivalence holds. The parameters fvs, fas and ddp are upper bounds to the other mentioned width measures (see Table 5.1). It is also much easier to compute these width parameters on directed co-graphs than on general digraphs. While the question, whether a digraph has at most width k concerning one of the mentioned parameters is NP-hard or even PSPACE-hard, d-pw, d-tw, cr and dagw are computable in linear time on directed co-graphs.

Many of these relations remain true for the superclass of directed co-graphs, which we call directed twin-distance-hereditary graphs [KR21]. These graphs are the directed version of the definition of distance-hereditary graphs by twins and pendant vertices. We could prove that for every twin-ddh digraph, all strong components are directed co-graphs and thus, we can generalize the results for all directed graph parameters, for which the directed width equals the maximum width of all strong components. It therefore holds that d-pw, d-tw, dagw and cr are equivalent on directed twin-ddh graphs and computable in linear time.

The equivalence of these parameters can also be generalized to extended directed co-graphs, but to compute them, we could only find polynomial time algorithms on this graph class right now.

On sequence digraphs, there is an XP-algorithm to compute directed path-width [GRR18, GRR21].

A further graph class to consider are semicomplete digraphs. In general directed clique-width is incomparable to all tree- and path-structure-based parameters. On directed co-graphs and directed twin-dh graphs, the directed clique-width is constant, which admits some interesting results, but rules out a comparison between d-cw and other parameters. On semicomplete digraphs however, there exists a result that directed path-width is an upper bound to directed clique-width [FP19], which we can extend to directed linear clique-width. This result permits to use Courcelle's Theorem for directed clique-width, i.e. the computability of all MSO_1-definable problems for graphs of bounded d-cw also for semicomplete graphs of bounded d-pw. In this work (and in [GKRW21]), we show equality between d-pw and dagw and equivalence between those parameters and d-tw and kw on semicomplete digraphs. By that, we can extend Courcelle's Theorem to all mentioned graph parameters on semicomplete digraphs.

Regarding all these results it gets clear that, although in general graph parameters for directed graphs seems to be of minor interest, on special digraph classes they could become very promising. In future work one could consider further directed graph classes. Another idea would be to investigate applied digraph problems, for example in bioinformatics, and see if the results in this work could help solve these digraph problems.

Bibliography

[ACP87] S. Arnborg, D.G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal of Algebraic and Discrete Methods*, 8(2):277–284, 1987.

[Adl07] I. Adler. Directed tree-width examples. *Journal of Combinatorial Theory, Series B*, 97(5):718–725, 2007.

[AH73] D. Adolphson and T.C. Hu. Optimal linear ordering. *SIAM Journal on Applied Mathematics*, 25(3):403–423, 1973.

[AH15] S.D. Andres and W. Hochstättler. Perfect digraphs. *Journal of Graph Theory*, 79(1):21–29, 2015.

[AKK+15] S.A. Amiri, L. Kaiser, S. Kreutzer, R. Rabinovich, and S. Siebertz. Graph Searching Games and Width Measures for Directed Graphs. In *32nd International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 30 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 34–47. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2015.

[AKK17] I. Adler, M.M. Kanté, and O. Kwon. Linear rank-width of distance-hereditary graphs I. A polynomial-time algorithm. *Algorithmica*, 78(1):342–377, 2017.

[AKR16] S. A. Amiri, S. Kreutzer, and R. Rabinovich. DAG-width is PSPACE-complete. *Theoretical Computer Science*, 655:78–89, 2016.

[Bar06] J. Barát. Directed pathwidth and monotonicity in digraph searching. *Graphs and Combinatorics*, 22:161–172, 2006.

[BdGR97] D. Bechet, P. de Groote, and C. Retoré. A complete axiomatisation of the inclusion of series-parallel partial orders. In *Rewriting Techniques and Applications*, volume 1232 of *LNCS*, pages 230–240. Springer-Verlag, 1997.

[BDH+12] D. Berwanger, A. Dawar, P. Hunter, S. Kreutzer, and J. Obdrzálek. The dag-width of directed graphs. *Journal of Combinatorial Theory, Series B*, 102(4):900–923, 2012.

[BDHK06] D. Berwanger, A. Dawar, P. Hunter, and S. Kreutzer. The dag-width and parity games. In *Proceedings of the Annual Symposium of Theoretical Aspects of Computer Science (STACS)*, volume 3884 of *LNCS*, pages 524–536. Springer-Verlag, 2006.

[BDK00] H.J. Broersma, E. Dahlhaus, and T. Kloks. A linear time algorithm for minimum fill-in and treewidth for distance hereditary graphs. *Discrete Applied Mathematics*, 99(1-3):367–400, 2000.

[BFJ+04] D. Bokal, G. Fijavz, M. Juvan, P.M. Kayll, and B. Mohar. The circular chromatic number of a digraph. *Journal of Graph Theory*, 46(3):227–240, 2004.

[BG18] J. Bang-Jensen and G. Z. Gutin, editors. *Classes of Directed Graphs*. Springer Monographs in Mathematics. Springer, 2018.

[BJG09] J. Bang-Jensen and G. Gutin. *Digraphs. Theory, Algorithms and Applications*. Springer-Verlag, Berlin, 2009.

[BJG18] J. Bang-Jensen and G. Gutin, editors. *Classes of Directed Graphs*. Springer-Verlag, Berlin, 2018.

[BKK93] H.L. Bodlaender, T. Kloks, and D. Kratsch. Treewidth and pathwidth of permutation graphs. In *Proceedings of International Colloquium on Automata, Languages and Programming (ICALP)*, volume 700 of *LNCS*, pages 114–125. Springer-Verlag, 1993.

[BLS99] A. Brandstädt, V.B. Le, and J.P. Spinrad. *Graph Classes: A Survey*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia, 1999.

[BM86] H.-J. Bandelt and H.M. Mulder. Distance-hereditary graphs. *Journal of Combinatorial Theory, Series B*, 41:182–208, 1986.

[BM90] H.L. Bodlaender and R.H. Möhring. The pathwidth and treewidth of cographs. In *Proceedings of Scandinavian Workshop on Algorithm Theory (SWAT)*, volume 447 of *LNCS*, pages 301–309. Springer-Verlag, 1990.

[BM93] H.L. Bodlaender and R.H. Möhring. The pathwidth and treewidth of cographs. *SIAM J. Disc. Math.*, 6(2):181–188, 1993.

[BMP20] B.M. Bumpus, K. Meeks, and W. Pettersson. Directed branch-width: A directed analogue of tree-width. *arXiv preprint arXiv:2009.08903*, 2020.

[Bod96] H.L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

[Bod98] H.L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998.

[Boe18] D. Boeckner. Oriented threshold graphs. *Australasian Journal of Combinatorics*, 71(1):43–53, 2018.

[BU84] M. Burlet and J.P. Uhry. Parity graphs. *Annals of Discrete Mathematics*, 21:253–277, 1984.

[CD06] J.-F. Culus and M. Demange. Oriented coloring: Complexity and approximation. In *Proceedings of the Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 3831 of *LNCS*, pages 226–236. Springer-Verlag, 2006.

[CER93] B. Courcelle, J. Engelfriet, and G. Rozenberg. Handle-rewriting hypergraph grammars. *Journal of Computer and System Sciences*, 46:218–270, 1993.

[CFS12] M. Chudnovsky, A.O. Fradkin, and P.D. Seymour. Tournament immersion and cutwidth. *Journal of Combinatorial Theory, Series B*, 102(1):93–101, 2012.

[CH77] V. Chvátal and P.L. Hammer. Aggregation of inequalities in integer programming. *Annals of Discrete Math.*, 1:145–162, 1977.

[CLMS19] V. Campos, R. Lopes, A.K. Maia, and I. Sau. Adapting the directed grid theorem into an fpt algorithm. *Electronic Notes in Theoretical Computer Science*, 346:229–240, 2019.

[CLSB81] D.G. Corneil, H. Lerchs, and L. Stewart-Burlingham. Complement reducible graphs. *Discrete Applied Mathematics*, 3:163–174, 1981.

[CMR00] B. Courcelle, J.A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory of Computing Systems*, 33(2):125–150, 2000.

[CO00] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101:77–114, 2000.

[Coh68] R.S. Cohen. Transition graphs and the star height problem. In *Proceedings of the 9th Annual Symposium on Switching and Automata Theory*, pages 383–394. IEEE Computer Society, 1968.

[Cou94] B. Courcelle. The monadic second-order logic of graphs VI: On several representations of graphs by relational structures. *Discrete Applied Mathematics*, 54:117–149, 1994.

[CP06] C. Crespelle and C. Paul. Fully dynamic recognition algorithm and certificate for directed cographs. *Discrete Applied Mathematics*, 154(12):1722–1741, 2006.

[CS11] M. Chudnovsky and P.D. Seymour. A well-quasi-order for tournaments. *Journal of Combinatorial Theory, Series B*, 101(1):47–53, 2011.

[DES14] M. Dehmer and F. Emmert-Streib, editors. *Quantitative Graph Theory: Mathematical Foundations and Applications*. Crc Pr Inc, New York, 2014.

[DS14] J. Dybizbański and A. Szepietowski. The oriented chromatic number of Halin graphs. *Information Processing Letters*, 114(1-2):45–49, 2014.

[Egg63] L.E. Eggan. Transition graphs and the star height of regular events. *Michigan Math. J.*, 10:385–397, 1963.

[EGW03] W. Espelage, F. Gurski, and E. Wanke. Deciding clique-width for graphs of bounded tree-width. *Journal of Graph Algorithms and Applications - Special Issue of JGAA on WADS 2001*, 7(2):141–180, 2003.

[EMC88] E. El-Mallah and C.J. Colbourn. The complexity of some edge deletion problems. *IEEE Transactions on Circuits and Systems*, 35(3):354–362, 1988.

[EST94] J.A. Ellis, I.H. Sudborough, and J.S. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113(1):50–79, 1994.

[FGL+18] F.V. Fomin, P. Golovach, D. Lokshtanov, S. Saurabh, and M. Zehavi. Cliquewidth III: The Odd Case of Graph Coloring Parameterized by Cliquewidth. *ACM Transactions on Algorithms*, 15(1):9:1–9:27, 2018.

[FHM03] T. Feder, P. Hell, and B. Mohar. Acyclic homomorphisms and circular colorings of digraphs. *SIAM Journal on Discrete Mathematics*, 17(1):161–163, 2003.

[FHP19] F. Foucaud, S. Heydarshahi, and A. Parreau. Domination and location in twin-free digraphs. *CoRR*, abs/1910.05311, 2019.

[FOT10] F.V. Fomin, S. Oum, and D. Thilikos. Rank-width and tree-width of H-minor-free graphs. *European Journal of Combinatorics*, 31(7):1617–1628, 2010.

[FP13a] F.V. Fomin and M. Pilipczuk. Jungles, bundles, and fixed parameter tractability. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 396–413. ACM-SIAM, 2013.

[FP13b] F.V. Fomin and M. Pilipczuk. Subexponential parameterized algorithm for computing the cutwidth of a semi-complete digraph. In *Proceedings of the Annual European Symposium on Algorithms (ESA)*, volume 8125 of *LNCS*, pages 505–516. Springer-Verlag, 2013.

[FP19] F.V. Fomin and M. Pilipczuk. On width measures and topological problems on semi-complete digraphs. *Journal of Combinatorial Theory, Series B*, 138:78–165, 2019.

[FRRS09] M.R. Fellows, F.A. Rosamond, U. Rotics, and S. Szeider. Clique-width is NP-complete. *SIAM Journal on Discrete Mathematics*, 23(2):909–939, 2009.

[FS13] A. Fradkin and P.D. Seymour. Tournament pathwidth and topological containment. *Journal of Combinatorial Theory, Series B*, 103:374–384, 2013.

[Gan11] R. Ganian. Thread graphs, linear rank-width and their algorithmic applications. In *Proceedings of International Workshop on Combinatorial Algorithms*, volume 6460 of *LNCS*, pages 38–42. Springer-Verlag, 2011.

[Gav77] F. Gavril. Some NP-complete problems on graphs. In *Proceedings of the 11th Conference on Information Sciences and Systems. Johns Hopkins University, Baltimore, Md.*, pages 91–95, 1977.

[GHK+09] R. Ganian, P. Hlinený, J. Kneis, A. Langer, J. Obdrzálek, and P. Rossmanith. On digraph width measures in parameterized algorithmics. In *Proceedings of the International Symposium on Parameterized and Exact Computation*, volume 5917 of *LNCS*, pages 185–197. Springer-Verlag, 2009.

[GHK+10] R. Ganian, P. Hlinený, J. Kneis, D. Meisters, J. Obdrzálek, P. Rossmanith, and S. Sikdar. Are there any good digraph width measures? In *Proceedings of the International Symposium on Parameterized and Exact Computation*, volume 6478 of *Lecture Notes in Computer Science*, pages 135–146. Springer-Verlag, 2010.

[GHK+14] R. Ganian, P. Hlinený, J. Kneis, A. Langer, J. Obdrzálek, and P. Rossmanith. Digraph width measures in parameterized algorithmics. *Discrete Applied Mathematics*, 168:88–107, 2014.

[GHK+16] R. Ganian, P. Hlinený, J. Kneis, D. Meisters, J. Obdrzálek, P. Rossmanith, and S. Sikdar. Are there any good digraph width measures? *Journal of Combinatorial Theory, Series B*, 116:250–286, 2016.

[GKL20] F. Gurski, D. Komander, and M. Lindemann. Oriented coloring of msp-digraphs and oriented co-graphs. In *International Conference on Combinatorial Optimization and Applications*, pages 743–758. Springer, 2020.

[GKR19a] F. Gurski, D. Komander, and C. Rehs. Computing digraph width measures on directed co-graphs. In *Proceedings of International Symposium on Fundamentals of Computation Theory (FCT)*, volume 11651 of *LNCS*, pages 292–305. Springer-Verlag, 2019.

[GKR19b] F. Gurski, D. Komander, and C. Rehs. Oriented coloring on recursively defined digraphs. *Algorithms*, 12(4):87, 2019.

[GKR21a] F. Gurski, D. Komander, and C. Rehs. Acyclic coloring parameterized by directed clique-width. In *Proceedings of the International Conference on Algorithms and Discrete Applied Mathematics (CALDAM)*, volume 12601 of *LNCS*, pages 95–108. Springer-Verlag, 2021.

[GKR21b] F. Gurski, D. Komander, and C. Rehs. How to compute digraph width measures on directed co-graphs. *Theoretical Computer Science*, 855:161–185, 2021.

[GKRW21] F. Gurski, D. Komander, C. Rehs, and S. Widerrecht. Directed width parameters on semicomplete digraphs. 2021. Submitted at COCOA.

[Gou12] R. Gould. *Graph Theory*. Dover Publications Inc., New York, NY, USA, 2012.

[GR99] M.C. Golumbic and U. Rotics. On the clique-width of some perfect graph classes. In *Proceedings of Graph-Theoretical Concepts in Computer Science (WG)*, volume 1665 of *LNCS*, pages 135–147. Springer-Verlag, 1999.

[GR00] M.C. Golumbic and U. Rotics. On the clique-width of some perfect graph classes. *International Journal of Foundations of Computer Science*, 11(3):423–443, 2000.

[GR18] F. Gurski and C. Rehs. Directed path-width and directed tree-width of directed co-graphs. In *Proceedings of the International Conference on Computing and Combinatorics (COCOON)*, volume 10976 of *LNCS*, pages 255–267. Springer-Verlag, 2018.

[GR19a] F. Gurski and C. Rehs. Comparing linear width parameters for directed graphs. *Theory of Computing Systems*, 63(6):1358–1387, 2019.

[GR19b] F. Gurski and C. Rehs. Forbidden directed minors, directed path-width and directed tree-width of tree-like digraphs. In *Proceedings of the Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 11376 of *LNCS*, pages 234–246. Springer-Verlag, 2019.

[GRR18] F. Gurski, C. Rehs, and J. Rethmann. Directed pathwidth of sequence digraphs. In *Proceedings of the International Conference on Combinatorial Optimization and Applications (COCOA)*, volume 11346 of *LNCS*, pages 79–93. Springer-Verlag, 2018.

[GRR21] F. Gurski, C. Rehs, and J. Rethmann. Characterizations and Directed Path-Width of Sequence Digraphs. In *Theory of Computing Systems*, 2021. to appear.

[Gru08] H. Gruber. Digraph complexity measures and applications in formal language theory. In *Proceedings of MEMICS'08*, pages 60–67, 2008.

[Gru12] H. Gruber. Digraph complexity measures and applications in formal language theory. *Discrete Mathematics and Theoretical Computer Science*, 14(2):189–204, 2012.

[GRW16] F. Gurski, J. Rethmann, and E. Wanke. On the complexity of the FIFO stack-up problem. *Mathematical Methods of Operations Research*, 83(1):33–52, 2016.

[Gur06a] F. Gurski. Characterizations for co-graphs defined by restricted NLC-width or clique-width operations. *Discrete Mathematics*, 306(2):271–277, 2006.

[Gur06b] F. Gurski. Linear layouts measuring neighbourhoods in graphs. *Discrete Mathematics*, 306(15):1637–1650, 2006.

[Gur08] F. Gurski. Graph parameters measuring neighbourhoods in graphs – Bounds and applications. *Discrete Applied Mathematics*, 156(10):1865–1874, 2008.

[Gur17] F. Gurski. Dynamic programming algorithms on directed cographs. *Statistics, Optimization and Information Computing*, 5:35–44, 2017.

[Gus93] J. Gusted. On the pathwidth of chordal graphs. *Discrete Applied Mathematics*, 45(3):233–248, 1993.

[GW00] F. Gurski and E. Wanke. The tree-width of clique-width bounded graphs without $K_{n,n}$. In *Proceedings of Graph-Theoretical Concepts in Computer Science (WG)*, volume 1938 of *LNCS*, pages 196–205. Springer-Verlag, 2000.

[GW05a] F. Gurski and E. Wanke. Minimizing NLC-width is NP-complete. In *Proceedings of Graph-Theoretical Concepts in Computer Science (WG)*, volume 3787 of *LNCS*, pages 69–80. Springer-Verlag, 2005.

[GW05b] F. Gurski and E. Wanke. On the relationship between NLC-width and linear NLC-width. *Theoretical Computer Science*, 347(1-2):76–89, 2005.

[GWY16] F. Gurski, E. Wanke, and E. Yilmaz. Directed NLC-width. *Theoretical Computer Science*, 616:1–17, 2016.

[GY02] G. Z. Gutin and A. Yeo. Orientations of digraphs almost preserving diameter. *Discrete Applied Mathematics*, 121(1-3):129–138, 2002.

[Hal76] R. Halin. S-functions for graphs. *J. Geometry*, 8:171–176, 1976.

[HK08] P. Hunter and S. Kreutzer. Digraph measures: Kelly decompositions, games, and orderings. *Theoretical Computer Science*, 399(3):206–219, 2008.

[HKK19] M. Hatzel, K. Kawarabayashi, and S. Kreutzer. Polynomial planar directed grid theorem. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1465–1484. ACM-SIAM, 2019.

[HMP09] P. Heggernes, D. Meister, and C. Papadopoulos. A complete characterisation of the linear clique-width of path powers. In *Proceedings of the Annual Conference on Theory and Applications of Models of Computation*, volume 5532 of *LNCS*, pages 241–250. Springer-Verlag, 2009.

[HMP11] P. Heggernes, D. Meister, and C. Papadopoulos. Graphs of linear clique-width at most 3. *Theoretical Computer Science*, 412(39):5466–5486, 2011.

[HOSG08] P. Hlinený, S. Oum, D. Seese, and G. Gottlob. Width parameters beyond tree-width and their applications. *Computer Journal*, 51(3):326–362, 2008.

[How77] E. Howorka. A characterization of distance-hereditary graphs. *The Quarterly Journal of Mathematics Ser. 2*, 28:417–420, 1977.

[HP05] M. Habib and C. Paul. A simple linear time algorithm for cograph recognition. *Discrete Applied Mathematics*, 145:183–197, 2005.

[HU53] F. Harary and G.E. Uhlenbeck. On the number of husimi trees: I. *Proceedings of the National Academy of Sciences*, 39(4):315–322, 1953.

[JKT21] L. Jaffke, O. Kwon, and J. A. Telle. Classes of intersection digraphs with good algorithmic properties. *ACM Computing Research Repository (CoRR)*, abs/2105.01413, 2021.

[Joh98] Ö. Johansson. Clique-decomposition, NLC-decomposition, and modular decomposition - relationships and results for random graphs. *Congressus Numerantium*, 132:39–60, 1998.

[JRST01a] T. Johnson, N. Robertson, P.D. Seymour, and R. Thomas. Addentum to "Directed tree-width", 2001.

[JRST01b] T. Johnson, N. Robertson, P.D. Seymour, and R. Thomas. Directed tree-width. *Journal of Combinatorial Theory, Series B*, 82:138–155, 2001.

[Jun78] H.A. Jung. On a class of posets and the corresponding comparability graphs. *Journal of Combinatorial Theory, Series B*, 24:125–133, 1978.

[Kar72] R.M. Karp. Reducibility among combinatorial problems. In R.E. Miller and J.W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103, New York, 1972. Plenum Press.

[Kas08] N. Kashyap. Matroid pathwidth and code trellis complexity. *SIAM J. Discrete Math.*, 22(1):256–272, 2008.

[KBMK93] T. Kloks, H. Bodlaender, H. Müller, and D. Kratsch. Computing treewidth and minimum fill-in: All you need are the minimal separators. In *Proceedings of the Annual European Symposium on Algorithms (ESA)*, volume 726 of *LNCS*, pages 260–271. Springer-Verlag, 1993.

[KF79] T. Kashiwabara and T. Fujisawa. NP-completeness of the problem of finding a minimum-clique-number interval graph containing a given graph as a subgraph. In *Proceedings of the International Symposium on Circuits and Systems*, pages 657–660, 1979.

[KK15] K. Kawarabayashi and S. Kreutzer. The directed grid theorem. In *Proceedings of the Annual ACM Symposium on Theory of Computing (STOC)*, pages 655–664. ACM, 2015.

[KKK+16] K. Kitsunai, Y. Kobayashi, K. Komuro, H. Tamaki, and T. Tano. Computing directed pathwidth in $O(1.89^n)$ time. *Algorithmica*, 75:138–157, 2016.

[KKT15] K. Kitsunai, Y. Kobayashi, and H. Tamaki. On the pathwidth of almost semicomplete digraphs. In *Proceedings of the Annual European Symposium on Algorithms (ESA)*, volume 9294 of *LNCS*, pages 816–827. Springer-Verlag, 2015.

[Kob15] Y. Kobayashi. Computing the pathwidth of directed graphs with small vertex cover. *Information Processing Letters*, 115(2):310–312, 2015.

[KR09] M. Kanté and M. Rao. Directed rank-width and displit decomposition. In *Proceedings of Graph-Theoretical Concepts in Computer Science (WG)*, volume 5911 of *LNCS*, pages 214–225. Springer-Verlag, 2009.

[KR13] M. Kanté and M. Rao. The rank-width of edge-coloured graphs. *Theory Comput. Syst.*, 52(4):599–644, 2013.

[KR21] D. Komander and C. Rehs. Twin-distance hereditary digraphs. 2021. arXiv preprint arXiv:2112.04183.

[KS15] I. Kim and P.D. Seymour. Tournament minors. *Journal of Combinatorial Theory, Series B*, 112(C):138–153, 2015.

[KZ15] S. Kintali and Q. Zhang. Forbidden directed minors and directed pathwidth. Research Report, 2015.

[KZ17] S. Kintali and Q. Zhang. Forbidden directed minors and Kelly-width. *Theoretical Computer Science*, 662:40–47, 2017.

[Law76] E.L. Lawler. Graphical algorithms and their complexity. *Math. Centre Tracts*, 81:3–32, 1976.

[Ler71] H. Lerchs. On cliques and kernels. Technical report, Dept. of Comput. Sci, Univ. of Toronto, 1971.

[LM17] Z. Li and B. Mohar. Planar digraphs of digirth four are 2-colorable. *SIAM J. Discrete Math.*, 31:2201–2205, 2017.

[LS10] M. Lätsch and R. Schrader. Distance-hereditary digraphs. *Journal of Discrete Algorithms*, 8(2):231–240, 2010.

[LT79] R.J. Lipton and R.E. Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[Mar13] T.H. Marshall. Homomorphism bounds for oriented planar graphs of given minimum girth. *Graphs and Combin.*, 29:1489–1499, 2013.

[McN69] R. McNaughton. The loop complexity of regular events. *Information Sciences*, 1(3):305–328, 1969.

[Moh03] B. Mohar. Circular colorings of edge-weighted graphs. *Journal of Graph Theory*, 43(2):107–116, 2003.

[MP95] N.V.R. Mahadev and U.N. Peled. *Threshold Graphs and Related Topics*. Annals of Discrete Math. 56. Elsevier, North-Holland, 1995.

[MRW20] N. Morawietz, C. Rehs, and M. Weller. A Timecop's Work Is Harder Than You Think. In *Proceedings of Mathematical Foundations of Computer Science (MFCS)*, volume 170 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 71:1–71:14, 2020.

[MS88] B. Monien and I.H. Sudborough. Min cut is NP-complete for edge weighted trees. *Theoretical Computer Science*, 58:209–229, 1988.

[MSW19] M.G. Millani, R. Steiner, and S. Wiederrecht. Colouring non-even digraphs. *ACM Computing Research Repository (CoRR)*, abs/1903.02872:37 pages, 2019.

[Nag12] H. Nagamochi. Linear layouts in submodular systems. In *Proceedings of the International Symposium on Algorithms and Computation*, volume 7676 of *LNCS*, pages 475–484. Springer-Verlag, 2012.

[NdM06] J. Nešetřil and P.O. de Mendez. Tree-depth, subgraph coloring and homomorphism bounds. *Eur. J. Comb.*, 27:1022–1041, 2006.

[NL82] V. Neumann-Lara. The dichromatic number of a digraph. *Journal of Combinatorial Theory, Series B*, 33(2):265–270, 1982.

[Obd06] J. Obdrzálek. Dag-width: Connectivity measure for directed graphs. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 814–821. ACM-SIAM, 2006.

[OS06] S. Oum and P.D. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006.

[Oum05] S. Oum. Rank-width and vertex-minors. *Journal of Combinatorial Theory, Series B*, 95:79–100, 2005.

[Oum17] S. Oum. Rank-width: Algorithmic and structural results. *Discrete Applied Mathematics*, 231:15–24, 2017.

[Ree99] B. Reed. Introducing directed tree width. *Electronic Notes in Discrete Mathematics*, 3:222–229, 1999.

[RS83] N. Robertson and P.D. Seymour. Graph minors I. Excluding a forest. *Journal of Combinatorial Theory, Series B*, 35:39–61, 1983.

[RS86a] N. Robertson and P.D. Seymour. Graph minors II. Algorithmic aspects of tree width. *Journal of Algorithms*, 7:309–322, 1986.

[RS86b] N. Robertson and P.D. Seymour. Graph minors V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41:92–114, 1986.

[RST94] N. Robertson, P.D. Seymour, and R. Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 62(2):323–348, 1994.

[Sch89] P. Scheffler. *Die Baumweite von Graphen als Mass für die Kompliziertheit algorithmischer Probleme.* Ph. D. thesis, Akademie der Wissenschaften in der DDR, Berlin, 1989.

[Sch21] R. Schrader. Personal communication, 2021.

[Sei11] S. Seidler. Über Minoren gerichteter Graphen. Diplomarbeit, TU Dresden, 2011.

[Sop97] É. Sopena. The chromatic number of oriented graphs. *Journal of Graph Theory*, 25:191–205, 1997.

[ST93] P.D. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *Journal of Combinatorial Theory, Series B*, 58(1):22–33, 1993.

[ST07] K. Suchan and I. Todinca. Pathwidth of circular-arc graphs. In *Proceedings of Graph-Theoretical Concepts in Computer Science (WG)*, volume 4769 of *LNCS*, pages 258–269. Springer-Verlag, 2007.

[Sum74] P.D. Sumner. Dacey graphs. *Journal of Aust. Soc.*, 18:492–502, 1974.

[SW20] R. Steiner and S. Wiederrecht. Parameterized algorithms for directed modular width. In *Proceedings of the International Conference on Algorithms and Discrete Applied Mathematics (CALDAM)*, volume 12016 of *LNCS*, pages 415–426. Springer-Verlag, 2020.

[Tam11] H. Tamaki. A Polynomial Time Algorithm for Bounded Directed Pathwidth. In *Proceedings of Graph-Theoretical Concepts in Computer Science (WG)*, volume 6986 of *LNCS*, pages 331–342. Springer-Verlag, 2011.

[Wan94] E. Wanke. k-NLC graphs and polynomial algorithms. *Discrete Applied Mathematics*, 54:251–266, 1994.

[Wie20] S. Wiederrecht. Digraphs of directed tree-width one. *Discrete Mathematics*, 343(12):112–124, 2020.

[YC08] B. Yang and Y. Cao. Digraph searching, directed vertex separation and directed pathwidth. *Discrete Applied Mathematics*, 156(10):1822–1837, 2008.

7 Appendix

7.1 Parts of my published Papers

Directed path-width and directed tree-width of directed co-graphs [GR18]

On this paper, I worked with my supervisor Frank Gurski. We did a lot of literature search together. While the results for directed path-width can be obtained very similar to the undirected case, what we did together, the results for directed tree-width are much more difficult. The main results and proofs for directed tree-width were my work.

Directed path-width of Sequence Digraphs [GRR18]

On this paper I worked with Frank Gurski and Jochen Rethmann. My main results in this paper are the characterizations of sequence digraph classes, especially Proposition 2 and Theorem 2.

Comparing linear width parameters for directed graphs [GR19a]

On this paper, I worked with Frank Gurski. We both generalized many results on undirected linear width parameters to directed linear width parameters. While he did all directed clique-width relations and I did some of the directed cut-width, directed rank-width and directed neighborhood-width relations. As this was my first paper (though it was published later, as it is a journal paper and not a conference paper), Frank Gurski revised many of my proofs.

Forbidden directed minors, directed path-width and directed tree-width of tree-like digraphs [GR19b]

On this paper, I worked nearly alone, with two small lemmas and some proof-reading of Frank Gurski.

Oriented coloring on recursively defined digraphs [GKR19b]

On this paper, I mainly worked with Frank Gurski. My at that time new colleague Dominque Komander only gave some minor support. Frank Gurski and me did a lot of literature search together. While he formulated most of the preliminaries, I created the two main algorithms of the paper by strongly modifying some undirected algorithms. Therefore, I invented the definition of a canonical di-co-tree. I also proved running time and correctness for these algorithms and some of the lemmas and corollaries. Only for the correctness proof of the second algorithm I worked with Dominique Komander.

Computing digraph width measures on directed co-graphs [GKR19a]

On this paper I worked with Frank Gurski and Dominique Komander. It shows how to compute different directed width measures on recursively defined graphs. While directed path-width and directed tree-width only cite [GR18], the results of DVFS-Number, Cycle Rank and DAG-depth were obtained by Dominique Komander and Frank Gurski. However, I developed the much more complex results for DAG-width. I further elaborated the results for Kelly-width in collaboration with Dominique Komander.

How to compute digraph width measures on directed co-graphs [GKR21b]

This paper is a long version of the papers [GR18] and [GKR19a]. My parts of this paper are the same as described above. In this paper, a mistake in a proof of [GR18] is corrected by several lemmata and a modification of this proof, which is primarily my work with some help from Dominique Komander.

Acyclic coloring parameterized by directed clique-width [GKR21a]

For this paper, which was written during my parental leave, I only did some minor support for the proofs and some proof-reading and corrections in the end.

Directed Versions of Distance-Hereditary Digraphs [KR21]

On this paper I mainly worked with Dominique Komander. It bases on discussions about directed twins with her, Frank Gurski and Van Bang Le. While Dominique Komander and me did the main definitions together, her part of the paper were the characterizations by forbidden subdigraphs and different properties and my part were the algorithms for directed width parameters on twin-dh digraphs.

Characterizations and Directed Path-Width of Sequence Digraphs [GRR21]

This paper is a long version of [GRR18]. My parts of the paper are the same as described above.

Directed Width Parameters on Semicomplete Digraphs [GKRW21]

This paper is a strong cooperation with my colleague Dominique Komander. For most proofs, we worked together, sometimes with some help of our supervisor Frank Gurski. In the last terms of the paper Sebastian Wiederrecht joined us and helped with the proof to compare directed path-width and directed tree-width. Together we optimized the result for directed path-width and Kelly-width we had before. He further gave some very valuable ideas for a restructuring of this paper.