Finding Linear Arrangements of Hypergraphs with Bounded Cutwidth in Linear Time

Thekla Hamm
Algorithms and Complexity Group, TU Wien, Vienna, Austria

Abstract

Cutwidth is a fundamental graph layout parameter. It generalises to hypergraphs in a natural way and has been studied in a wide range of contexts. For graphs it is known that for a fixed constant k there is a linear time algorithm that for any given G, decides whether G has cutwidth at most k and, in the case of a positive answer, outputs a corresponding linear arrangement. We show that such an algorithm also exists for hypergraphs.

2012 ACM Subject Classification Mathematics of computing → Permutations and combinations; Mathematics of computing → Hypergraphs; Theory of computation → Dynamic graph algorithms

Keywords and phrases Fixed parameter linear, Path decomposition, Hypergraph

Digital Object Identifier 10.4230/LIPIcs.IPEC.2019.20

Funding Supported by the Austrian Science Fund (FWF, Project P31336 and Project W1255-N23).

1 Overview

For a hypergraph H and an enumeration of its vertices $V(H)$, also referred to as *linear arrangement*, the *cutwidth* is the smallest integer k such that for every position i between 1 and $|V(H)| - 1$ there are at most k hyperedges with one endpoint among the first i and the other among the last $|V(H)| - i$ vertices of the arrangement. Finding orderings with small cutwidth for hypergraphs naturally occurs directly in various applications, probably the most classical one being VLSI design [6]. Also, there are problems for which a vertex ordering with small cutwidth can be used to obtain a provably good processing order for heuristic approaches, such as SAT [15]. We want to emphasise that a model using hypergraphs, as opposed to graphs, is necessary in these scenarios among others, as graphs do not capture all dependencies correctly.

If one wants to find an ordering with smallest possible cutwidth, this is the classical Minimum Cut Linear Arrangement Problem which is known to be NP-hard, even on graphs with maximum vertex degree 3 [7]. However, if one considers a bound k on the maximum allowed cutwidth as a parameter, one can decide in linear time if a linear ordering of a hypergraph is possible such that the bound is not exceeded [14]. The known proof is non-constructive in the sense that it does not infer a way to construct such a linear ordering in linear time. The asymptotically fastest constructive algorithm given in literature [8] is not even FPT (fixed parameter tractable) as the runtime complexity lies in $\mathcal{O}(n^{k^2+3k+3})$ where n is the number of vertices of the given hypergraph.

This Work. We give a linear time constructive algorithm.

A key observation is that the fixed bound also bounds the pathwidth of the incidence graph of the hypergraph. This allows to compute a path decomposition of the incidence graph in linear time using the result of Bodlaender [1], and then to dynamically compute a solution along the path decomposition. This approach has been successfully applied to a number of problems. An overview of the general method for the slightly more general notion of tree decompositions, as well as some examples for classical problems for which it was used, can be found e.g. in [3, Chapter 7]. In particular, the method has also been used to give a

linear time algorithm for the problem we consider restricted to graphs [11]. We extend the crucial ideas of this algorithm.

We will construct a linear ordering by successively inserting vertices into it as long as the cutwidth bound is satisfied and successively take into consideration more and more of the hyperedges when checking whether the cutwidth bound is satisfied. It is reasonable that if one takes this approach, after a vertex and all its incident hyperedges have been completely processed the position of this vertex is no longer interesting as it has no further impact on yet to be processed vertices and hyperedges. Similarly if a hyperedge and all its vertices have been processed the hyperedge will not play a role when extending the incomplete linear ordering. The first statement can be strengthened using the fact that only the positions of the outermost vertices of each hyperedge are necessary to compute the cutwidth of a linear ordering: After a vertex has been processed and the positions of the left- and rightmost vertices with respect to the incomplete linear ordering of each incident hyperedge are known, the vertex will not play a role when extending the incomplete linear ordering. A path decomposition of the incidence graph implies some order of processing the vertices and hyperedges such that no more than width-of-the-decomposition-many vertices and hyperedges are relevant in the way described above at the same time.

Structure. We start by giving a formal definition of our problem and an overview over previous research in Section 2. In Section 3, we introduce the pathwidth of the incidence graph of a hypergraph, and prove that a bound on the cutwidth induces a bound on it. We also work to obtain a path decomposition that has convenient structural properties for our dynamic programming approach. Section 4 is dedicated to problem specific arguments: In Section 4.1 we prove that it is sufficient to consider linear arrangements that are constructed in a certain way along the path decomposition. The idea is the same that was used for graphs [11]. It utilises the notion of so called typical sequences. Using the results of the previous sections, we formulate a dynamic program that works on path decompositions of incidence graphs and allows us to present our main result in Section 4.2. In Section 5 we indicate a possibility to lift the algorithm to a more general setting.

2 Formal Introduction and Preliminary Observations

We start with a hypergraph $H = (V, E)$. That is V is some finite set and E contains subsets of vertices, possibly with multiplicities, i.e. we allow parallel hyperedges.

▶ **Definition 1.** *A **linear arrangement** of a hypergraph H is a bijection $\varphi : V(H) \leftrightarrow \{1, \dots, |V(H)|\}$. The **cutwidth** of a linear arrangement φ **at position** $i \in \mathbb{N}$ is defined as* $\mathbf{cw(\varphi, i)} = |\{e \in E(H) \mid \exists v, w \in e \quad \varphi(v) \leq i < \varphi(w)\}|$. *The **cutwidth** of a linear arrangement φ is defined as* $\mathbf{cw(\varphi)} = \max_{i \in \mathbb{N}} \mathrm{cw}(\varphi, i)$. *The **cutwidth** of a hypergraph H is defined as* $\mathbf{cw(H)} = \min_{\substack{\varphi \text{ linear} \\ \text{arrangement of } H}} \mathrm{cw}(\varphi)$.

If one is interested in the cutwidth of hypergraphs, one can neglect all hyperedges in H that contain less than two vertices, as such hyperedges do not contribute to the cutwidth of any linear arrangement of H. From now on we will assume that $\forall e \in E(H)$, $|e| \geq 2$ for our input hypergraph H and refer to this assumption as **hyperedge cardinality assumption**. Note that any hypergraph H' can be transformed to satisfy this condition in time in $\mathcal{O}(|E(H')|)$ by deleting all hyperedges that are too small.

k-Cutwidth Bounded Linear Arrangement (k-CWLA)	
Instance	Hypergraph $H = (V, E)$ such that $\forall e \in E(H)$ $\lvert e\rvert \geq 2$
Task	Find a linear arrangement φ of H with $\mathrm{cw}(\varphi) \leq k$ or decide that $\mathrm{cw}(H) > k$.

We present a linear time algorithm, running in time in $\mathcal{O}(|V(H)|)$ that solves k-CWLA. If one drops the condition on the hyperedge cardinalities, this directly implies an algorithm that runs in time in $\mathcal{O}(|E(H)| + |V(H)|)$.

2.1 Known Results

The best algorithm given in literature for k-CWLA runs in time in $\mathcal{O}(|V(H)|^{k^2+3k+3})$ [8] and relies on complicated dynamic programming on a linear arrangement which is iteratively extended. There is an FPT-algorithm for constructing path decompositions of polymatroids with bounded width. Concretely:

▶ **Theorem 2** ([4]). *Let $\mathbb{F}$ be a fixed finite field. Given n subspaces of $\mathbb{F}^r$ for some r and a parameter k, in time in $\mathcal{O}(rm^2 + n^3)$, we can either find an enumeration $V_1, V_2, \ldots, V_n$ of the subspaces, such that $\dim((V_1 + \ldots + V_i) \cap (V_{i+1} + \ldots + V_n)) \leq k$ for all $i \in \{1, \ldots, n-1\}$, or decide that no such enumeration exists, where each V_i is given by its spanning set of d_i vectors and $m = \sum_{i=1}^{n} d_i$.*

One can write k-CWLA in a way that this theorem is applicable by setting $\mathbb{F} = E(H)$ and the subspaces to be the $|V(H)|$ subspaces spanned by the hyperedges incident to each of the vertices of H. Then one obtains a $\mathcal{O}(|E(H)|^5 + |V(H)|^3)$ (or $\mathcal{O}(|E(H)|^3 + |V(H)|^3)$ using the same observations as we do in Section 2.2) algorithm to solve k-CWLA. Thus this approach yields a runtime which is not linear in the size of the hypergraph. Also, as it is not specifically adapted to the problem at hand but rather general, if one wants to understand the algorithmic details, it is comparatively complicated and probably impractical. We also remark that the algorithm presented in this paper, leaves scope for extending to a more general setting (see Section 5).

The corresponding decision problem to k-CWLA however, i.e. deciding if the cutwidth of a hypergraph is smaller than k without outputting a corresponding linear arrangement, is known to be solvable in time in $\mathcal{O}(|V(H)| + |E(H)|)$. This was proved in [14]. The given algorithm is non-constructive and uses an adaptation of the analogue of the Myhill-Nerode theorem for coloured graphs to work on the incidence graph of hypergraphs. If one restricts the problem to hypergraph instances for which the hyperedge cardinality assumption holds, as we do, one can use the same observations that we will make in Section 2.2 to obtain an algorithm that runs in $\mathcal{O}(|V(H)|)$. In [13] van Bevern restates the open problem of constructing a linear arrangement with k-bounded cutwidth if possible in linear time and points out the importance of such an algorithm for practical purposes. It is also worth noting that the comparatively heavy machinery used to obtain the decision result leads to a high dependency of the complexity on k, and obstructs combinatorial properties of a solution.

For k-CWLA restricted to graphs a linear time algorithm is known [11]. Our algorithm for hypergraphs uses and extends the crucial ideas of this algorithm.

2.2 The Vertex Degree Property

▶ **Lemma 3.** *Let H satisfy the hyperedge cardinality assumption. If for any $v \in V(H)$ we have that $|\{e \in E(H) \mid v \in e\}| > 2k$ then $\mathrm{cw}(H) > k$.*

Lemma 3 motivates considering the **vertex degree property** of a hypergraph H which H satisfies if $\forall v \in V(H)\ |\{e \in E(H) \mid v \in e\}| \leq 2k$. Obviously, if H satisfies the vertex degree property, $|E(H)| \in \mathcal{O}(|V(H)|)$.

When claiming runtime in $\mathcal{O}(|V(H)|)$, we need to make clear how we assume H to be given as input. Note that in general the analogue of an incidence matrix for hypergraphs does not allow to check the vertex degree property in linear time. However a natural analogue of an adjacency list for hypergraphs, in which we list for each vertex the hyperedges it is incident to, can easily be seen to allow the following:

▶ Remark 4.

- Checking if H satisfies the vertex degree property in time in $\mathcal{O}(|V(H)|)$,
- and if H satisfies the vertex degree property, checking the membership of a given vertex in a given hyperedge in constant time.

3 Path Decompositions of Incidence Graphs

The cutwidth of a hypergraph relates to the pathwidth of its incidence graph, in two ways: Firstly this parameter is bounded by the cutwidth and secondly its boundedness allows the decomposition of the hypergraph given as instance in a way that is useful for our algorithm. We will elaborate on the properties that make the decomposition useful in Section 4 and for now focus on the definition of the parameter, proving the fact that it is bounded by cutwidth and showing that there is a decomposition of our hypergraph with certain properties.

▶ **Definition 5.** *The* ***incidence graph*** $G_I(H)$ *of* H *is given by* $V(G_I(H)) = \{v_u \mid u \in V(H)\} \dot\cup \{v_e \mid e \in E(H)\}$ *and* $E(G_I(H)) = \{\{v_u, v_e\} \mid u \in e\}$.

For a thorough description of our algorithm, we need the following technical statement that allows us to transform our input into easily accessible information about the incidence graph in linear time. The proof is not difficult.

▶ **Lemma 6.** *Let* H *be a hypergraph that has the vertex degree property. The following can be computed in time in* $\mathcal{O}(|V(H)|)$ *from our representation of* H*:*

- *The representation of* $G_I(H)$ *by its adjacency list;*
- *for each* $x \in V(G_I(H))$, $\text{type}(x) \in \{$*'vertex'*, *'hyperedge'*$\}$ *with*

$$\text{type}(x) = \begin{cases} \text{'vertex'} & \text{if } x \in \{v_u \in G_I(H) \mid u \in V(H)\} \\ \text{'hyperedge'} & \text{if } x \in \{v_e \in G_I(H) \mid e \in E(H)\} \end{cases}; \text{ and}$$

- *for each* $x \in \{v_u \in V(G_I(H)) \mid u \in V(H)\}$ *a lookup table for the corresponding* $u \in V(H)$.

The following definition was introduced by Robertson and Seymour [10]. It is a specialisation of the notion of tree decomposition and treewidth. Both have been studied in various contexts, one being as foundation for dynamic parameterised algorithms [3, Chapter 7].

▶ **Definition 7.** *A* ***path decomposition*** *of a graph* $G = (V, E)$ *is a sequence* $(X_1, \ldots, X_s)$ *with* $X_i \subseteq V(G)$ *such that*

(i) $\forall e \in E(G)\ \exists 1 \leq i \leq s$ *such that* $e \subseteq X_i$*; and*

(ii) $\forall v \in V(G)\ \exists 1 \leq i \leq j \leq s$ *such that* $v \in X_k \Leftrightarrow i \leq k \leq j$.

The ***width*** *of a path decomposition* $(X_1, \ldots, X_s)$ *is defined as* $\mathbf{width}((X_1, \ldots, X_s)) = \max_{1 \leq i \leq s} |X_i| - 1$. *The* ***pathwidth*** *of a graph* G *is defined as minimum width of a path decomposition of* G*, i.e.* $\mathbf{pw}(G) = \min_{\substack{(X_1,\ldots,X_s) \\ \text{path decomposition of } G}} \text{width}((X_1, \ldots, X_s))$. *The* X_i *are called the* ***bags*** *of the path decomposition.*

▶ **Theorem 8.** *Let H satisfy the hyperedge cardinality assumption. If* $\mathrm{cw}(H) \leq k$ *then* $\mathrm{pw}(G_I(H)) \leq k$.

Proof. Let φ be a linear arrangement of H with $\mathrm{cw}(\varphi) \leq k$ and define for $i = 1, \ldots, |V(H)|$
$X_{2i-1} = \{v_e \mid \exists v, w \in e \quad \varphi(v) < i \leq \varphi(w)\} \cup \{v_{\varphi^{-1}(i)}\}$ and
$X_{2i} = \{v_e \mid \exists v, w \in e \quad \varphi(v) \leq i < \varphi(w)\} \cup \{v_{\varphi^{-1}(i)}\}$.
It is easy to verify that this defines a path decomposition of $G_I(H)$ with width at most k. ◀

The boundedness of pathwidth in connection with the following well-known result will allow us to find a path decomposition of the incidence graph with bounded width in linear time.

▶ **Theorem 9** ([1])**.** *Given some graph G, it can be decided in time in $\mathcal{O}(|V(G)|)$ if* $\mathrm{pw}(G) \leq k$ *and if this is the case a path decomposition $(X_1, \ldots, X_s)$ of G with width at most k and $s \in \mathcal{O}(|V(G)|)$ can be computed in time in $\mathcal{O}(|V(G)|)$.*

The remainder of this section is dedicated to transforming the path decomposition we obtain from Theorem 9 into a more convenient form. The first step is standard.

▶ **Definition 10.** *A path decomposition $(X_1, \ldots, X_s)$ is* ***nice*** *if for all $1 \leq i \leq s$ it holds that $|X_i \triangle X_{i-1}| = 1$. (We artificially set $X_0 = \emptyset$.) Then for every X_i one of the following holds:*

- *$|X_i \setminus X_{i-1}| = 1$, then call X_i* ***introduce bag****. We say x with $\{x\} = X_i \setminus X_{i-1}$ is* ***introduced****.*
- *$|X_{i-1} \setminus X_i| = 1$, then call X_i* ***forget bag****. We say x with $\{x\} = X_{i-1} \setminus X_i$ is* ***forgotten****.*

▶ **Lemma 11** ([2])**.** *A path decomposition $(X_1, \ldots, X_s)$ of a graph G can be transformed into a nice path decomposition $(X'_1, \ldots, X'_{s'})$ of G in time in $\mathcal{O}(s \cdot \mathrm{width}((X_1, \ldots, X_s)))$ without increasing the width and with $s' \in \mathcal{O}(s \cdot \mathrm{width}((X_1, \ldots, X_s)))$.*

The fact that we also need to handle hyperedges leads to some additional technicalities. For this reason we will want to use a nice path decomposition with an additional restriction, which we call *extra niceness*, in our dynamic program.

▶ **Definition 12.** *A path decomposition $(X_1, \ldots, X_s)$ of an incidence graph $G_I(H)$ is* ***extra nice*** *if it is nice and for all $e \in E(H)$ there is some $u \in e$ such that v_u appears in the path decomposition before v_e, i.e.* $\forall e \in E(H) \quad \min_{\substack{v_u \in X_i \\ u \in e}} i < \min_{v_e \in X_i} i$.

Just like path decompositions can be efficiently be transformed into nice path decompositions, it is easy to modify nice path decompositions of an incidence graph to obtain extra nice path decompositions efficiently.

▶ **Lemma 13.** *Let H satisfy the hyperedge cardinality assumption and have the vertex degree property. A nice path decomposition $(X_1, \ldots, X_s)$ of $G_I(H)$ can be transformed into an extra nice path decomposition $(X'_1, \ldots, X'_s)$ of $G_I(H)$ in time in $\mathcal{O}(s \cdot \mathrm{width}((X_1, \ldots, X_s))^2)$ without increasing the width.*

The results of this section applied in series prove the following theorem:

▶ **Theorem 14.** *Given a hypergraph H that satisfies the hyperedge cardinality assumption and has the vertex degree property it can be decided in time in $\mathcal{O}(|V(H)|)$ if* $\mathrm{pw}(G_I(H)) \leq k$ *and, if this is the case, an extra nice path decomposition $(X_1, \ldots, X_s)$ of $G_I(H)$ with width at most k and $s \in \mathcal{O}(|V(H)|)$ can be computed in time in $\mathcal{O}(|V(H)|)$.*

After the observations in this section we may assume to have an extra nice path decomposition $(X_1, \ldots, X_s)$ of $G_I(H)$ with $s \in \mathcal{O}(|V(H)|)$ and $\mathrm{width}((X_1, \ldots, X_s)) \leq k$.

4 A Linear Time Algorithm for k-CWLA

Using our path decomposition we will dynamically construct a linear arrangement with bounded cutwidth, working on hypergraphs H_i that grow along the path decomposition and are contained in H in a certain sense.

▶ **Definition 15.** *For a hypergraph H and some $W \subseteq V(H)$ and $F \subseteq E(H)$ the* ***trimmed subhypergraph*** *of H* ***induced by W and F*** *is defined as $\boldsymbol{H[W,F]}$ with vertex set $V(H[W,F]) = W$ and hyperedge set $E(H[W,F]) = \{e \cap W \mid e \in F\}$. We call a hypergraph H'* ***trimmed subhypergraph*** *of H if there are $W \subseteq V(H)$ and $F \subseteq E(H)$ such that $H' = H[W,F]$.*

▶ **Definition 16.** *For a linear arrangement φ of $H = (V,E)$ and some $H' = (V',E')$ trimmed subhypergraph of H define the* ***restriction*** *of φ to H' $\boldsymbol{\varphi|_{H'}}$ to be the unique linear arrangement of H' such that $\forall v,w \in V(H') \quad \varphi(v) \leq \varphi(w) \Leftrightarrow \varphi|_{H'}(v) \leq \varphi|_{H'}(w)$.*
Correspondingly, if ψ is a linear arrangement of $H = (V,E)$ and φ is the restriction of ψ to some $H' = (V',E')$ where $V' \subseteq V$ and $E' = \{e \cap V' \mid e \in E''\}$ where $E'' \subseteq E$, then ψ is an ***extension*** *of φ. For a vertex $v \in V(H) \setminus V(H')$ and $1 \leq p < |V(H')|$, we say that v is* ***inserted*** *into position p of φ by ψ, if $\psi(v) \in \{\psi(\varphi^{-1}(p)), \ldots, \psi(\varphi^{-1}(p+1))\}$, and that v is inserted into position 0 os φ by ψ if $\psi(v) < \psi(\varphi^{-1}(1))$, and into position $|V(H')|$ of φ by ψ if $\psi(v) > \psi(\varphi^{-1}(|V(H')|))$.*

▶ Remark 17. Note that restrictions are indeed well-defined as a linear arrangement of $V(H)$ infers a unique linear arrangement of any subset of $V(H)$.

We are interested specifically in the following trimmed subhypergraphs of H, along which we will later iteratively extend our linear arrangement.

▶ **Definition 18.** *For $1 \leq i \leq s$ define*
$\boldsymbol{H_i} = H[\{u \in V(H) \mid v_u \in \bigcup_{1 \leq j \leq i} X_j\}, \{e \in E(H) \mid v_e \in \bigcup_{1 \leq j \leq i} X_j\}]$, *and* $\boldsymbol{H_0} = (\emptyset, \emptyset)$.

Assume to be at stage $1 \leq i \leq s$ when traversing the path decomposition, and let φ be a linear arrangement of H_i that we want to extend. The vertices of H that are represented in X_i, as well as the outermost vertices of (possibly trimmed) hyperedges that are represented in X_i, are of particular interest. This is due to the fact that, by the properties of a path decomposition, these are the vertices that may be important in a certain sense, when updating information about φ to extensions that include parts of the hypergraph that are still to be encountered. We formalise this intuition in the following definition and characterising remark.

▶ **Definition 19.** *We say a vertex $v \in V(H_i)$ is* ***unimportant after $\boldsymbol{\varphi}$*** *if*

- *all hyperedges of H, incident to v are considered in $E(H_i)$, i.e. $\forall e \in E(H) \quad v \in e \Rightarrow e \cap V(H_i) \in E(H_i)$; and*
- *no hyperedge of H, containing vertices from $V(H_i)$, as well as $V(H) \setminus V(H_i)$, has any v as left- or rightmost vertex in φ, i.e.*
  $\forall e \in E(H) \quad e \cap V(H_i) \neq \emptyset \wedge e \cap \big(V(H) \setminus V(H_i)\big) \neq \emptyset$
  $$\Rightarrow \operatorname*{argmin}_{w \in e \cap V(H_i)} \varphi(w) \neq v \wedge \operatorname*{argmax}_{w \in e \cap V(H_i)} \varphi(w) \neq v.$$

Let $1 \leq i \leq s$ and φ be a linear arrangement of H_i. The set of ***vertices distinguished by $\boldsymbol{\varphi}$*** *is the set $\{u \in V(H) \mid v_u \in X_i\} \cup \{\operatorname*{argmin}_{u \in e \cap V(H_i)} \varphi(u), \operatorname*{argmax}_{u \in e \cap V(H_i)} \varphi(u) \mid v_e \in X_i\}$.*

▶ Remark 20. Note that by the properties of a path decomposition in particular all vertices in $V(H_i) \setminus \{u \in V(H) \mid v_u \in X_i\} \cup \{\operatorname{argmin}_{u \in e \cap V(H_i)} \varphi(u), \operatorname{argmax}_{u \in e \cap V(H_i)} \varphi(u) \mid v_e \in X_i\}$ are unimportant after φ, i.e. all vertices that are not distinguished by φ, are unimportant after φ.

The properties in the definition of unimportantness are what we will exploit to identify linear arrangements that we have need to consider at a certain stage, in terms of whether or not they allow an extension to a complete linear arrangement of H with cutwidth at most k. In this way we will be able to restrict ourselves to considering at most constantly many (w.r.t. the size of the hypergraph) linear arrangements at each stage. We give the details in Section 4.1 and use these results in Section 4.2 to give a linear time algorithm for k-CWLA.

4.1 Identifying Partial Linear Arrangements

We use the same idea as in [11] to give a condition under which two linear arrangements of H_i for some $i \in \{1, \ldots, s\}$ either can both, or can both not be extended to a linear arrangement of H satisfying the cutwidth bound. More specifically we consider the relative order of distinguished vertices in such linear arrangements and where exactly, vertices that are yet to be encountered may be inserted. In certain cases we will be able to shift vertices that are not distinguished and are strewn between vertices distinguished by φ between two distinguished vertices that are consecutive w.r.t. φ without increasing the cutwidth.

▶ **Lemma 21.** *Let $1 \leq i \leq s$ and φ be a linear arrangement of H_i. Assume there are p and $q \in \mathbb{N}$ such that*

(i) $\operatorname{cw}(\varphi, p) = \min_{0 \leq j \leq q} \operatorname{cw}(\varphi, p+j)$ *and* $\operatorname{cw}(\varphi, p+q) = \max_{0 \leq j \leq q} \operatorname{cw}(\varphi, p+j)$*; and*

(ii) $\varphi^{-1}(p+1), \ldots, \varphi^{-1}(p+q)$ *are unimportant after φ.*

Let ψ be a linear arrangement of H that extends φ. Obtain ψ' from ψ by shifting for each $1 \leq \bar{j} \leq q$ the vertices from in between $\varphi^{-1}(p+\bar{j})$ and $\varphi^{-1}(p+\bar{j}+1)$ to in between $\varphi^{-1}(p)$ and $\varphi^{-1}(p+1)$ and otherwise maintaining the same relative ordering as given by ψ. Formally the described ψ' is defined by:

$$\psi'(v) = \begin{cases} \psi(v) - \bar{j} & \text{if } \psi(\varphi^{-1}(p+\bar{j})) < \psi(v) < \psi(\varphi^{-1}(p+\bar{j}+1)) \\ & \quad \text{for some } 1 \leq \bar{j} \leq q \\ \psi(\varphi^{-1}(p+q+1)) - (q+1-\bar{j}) & \text{if } v = \varphi^{-1}(p+\bar{j}) \text{ for some } 1 \leq \bar{j} \leq q \\ \psi(v) & \text{otherwise} \end{cases}$$

Then $\operatorname{cw}(\psi') \leq \operatorname{cw}(\psi)$.

▶ Remark 22. See Figure 1 for an illustration of how φ, ψ and ψ' relate.

Proof. Note that if $j \notin \{\psi(\varphi^{-1}(p+1)), \ldots, \psi(\varphi^{-1}(p+q+1)) - 1\}$ by construction of ψ' it holds that $\forall v \in V(H) \quad \psi(v) \leq j \Leftrightarrow \psi'(v) \leq j$. So for all such j we have $\operatorname{cw}(\psi, j) = \operatorname{cw}(\psi', j)$. It remains to consider positions between $\psi(\varphi^{-1}(p+1))$ and $\psi(\varphi^{-1}(p+q+1)) - 1$. For $j' \in \{\psi(\varphi^{-1}(p+1)), \ldots, \psi(\varphi^{-1}(p+2)) - 1\}$ we will find $j \in \mathbb{N}$ such that $\operatorname{cw}(\psi', j') \leq \operatorname{cw}(\psi, j)$. We distinguish two cases according to the cases considered in the definition of the value of ψ' at $\psi'^{-1}(j')$:

Case 1: $\psi(\varphi^{-1}(p+\bar{j})) < \psi(\psi'^{-1}(j')) < \psi(\varphi^{-1}(p+\bar{j}+1))$ **for some** $1 \leq \bar{j} \leq q$ **.** One can set $j = \psi(\psi'^{-1}(j'))(= j' + \bar{j})$ (see Figure 1, red), and use the fact that $\varphi^{-1}(p+1), \ldots, \varphi^{-1}(p+q)$ are unimportant after φ, and the fact that $\operatorname{cw}(\varphi, p) \leq \operatorname{cw}(\varphi, p+\bar{j})$ to show $\operatorname{cw}(\psi', j') \leq \operatorname{cw}(\psi, j)$.

Case 2: $\psi'^{-1}(j') = \varphi^{-1}(p+\bar{j})$ **for some** $1 \leq \bar{j} \leq q$**.** One can set $j = \psi(\varphi^{-1}(p+q))$ (see Figure 1, blue), and use the fact that $\varphi^{-1}(p+1), \ldots, \varphi^{-1}(p+q)$ are unimportant after φ, and the fact that $\operatorname{cw}(\varphi, p+\bar{j}) \leq \operatorname{cw}(\varphi, p+q)$ to show $\operatorname{cw}(\psi', j') \leq \operatorname{cw}(\psi, j)$. ◀

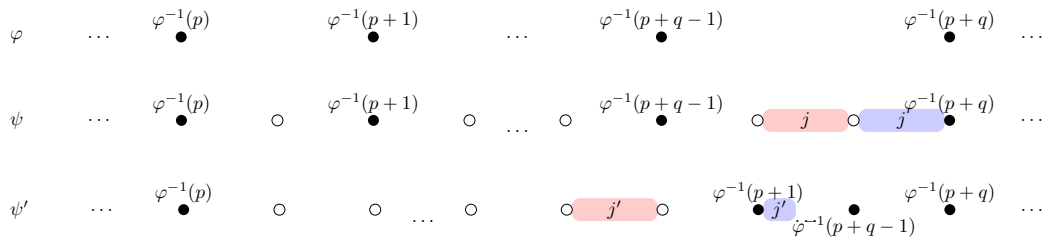

Figure 1 Illustrations of the situation in Lemma 21 and its proof – vertices in $V(H_i)$ are filled, exemplary positions at which cutwidths are compared in Case 1 red and Case 2 blue.

One can show an analogous result for the smallest cutwidth being at the largest position:

▶ **Lemma 23.** *Let $1 \leq i \leq s$ and φ be a linear arrangement of H_i. Assume there are p and $q \in \mathbb{N}$ such that*

(i) $\mathrm{cw}(\varphi, p+q) = \min_{0 \leq j \leq q} \mathrm{cw}(\varphi, p+j)$ *and* $\mathrm{cw}(\varphi, p) = \max_{0 \leq j \leq q} \mathrm{cw}(\varphi, p+j)$*; and*

(ii) $\varphi^{-1}(p+1), \ldots, \varphi^{-1}(p+q)$ *are unimportant after φ.*

Let ψ be a linear arrangement of H that extends φ. Obtain ψ' from ψ by shifting for each $0 \leq \bar{j} \leq q-1$ the vertices from in between $\varphi^{-1}(p+\bar{j})$ and $\varphi^{-1}(p+\bar{j}+1)$ to in between $\varphi^{-1}(p+q)$ and $\varphi^{-1}(p+q+1)$ and otherwise maintaining the same relative ordering as given by ψ. Then $\mathrm{cw}(\psi') \leq \mathrm{cw}(\psi)$.

Lemma 21 and Lemma 23 yield:

▶ **Lemma 24.** *Let $1 \leq i \leq s$ and φ be a linear arrangement of H_i.*

1. *If $p \in \{1, \ldots, |V(H_i)| - 1\}$ is a position, $\varphi^{-1}(p+1)$ is not a vertex distinguished by φ, and $\mathrm{cw}(\varphi, p) = \mathrm{cw}(\varphi, p+1)$, then there is a linear arrangement ψ' of H that extends φ such that $\mathrm{cw}(\psi') = \min_{\substack{\psi \text{ extension of} \\ \varphi \text{ to } H}} \mathrm{cw}(\psi)$, and no vertex is inserted into position $p+1$ of φ by ψ'.*
2. *If $q \geq 2$ and $p \in \{1, \ldots, |V(H_i)| - 1 - q\}$ is a position, for all $1 \leq j \leq q$, $\varphi^{-1}(p+j)$ is not a vertex distinguished by φ, and $\min_{0 \leq j \leq q} \mathrm{cw}(\varphi, p+j) = \mathrm{cw}(\varphi, p)$ and $\max_{0 \leq j \leq q} \mathrm{cw}(\varphi, p+j) = \mathrm{cw}(\varphi, p+q)$ or vice versa, then there is a linear arrangement ψ' of H that extends φ such that $\mathrm{cw}(\psi') = \min_{\substack{\psi \text{ extension of} \\ \varphi \text{ to } H}} \mathrm{cw}(\psi)$, and no vertex is inserted into any of the positions $p+1, \ldots, p+q-1$ of φ by ψ'.*

Proof. Apply Lemma 21 or Lemma 23 to ψ and the relevant positions, where ψ is any linear arrangement of H that extends φ and achieves minimal cutwidth among these. ◀

This immediately relates to so called typical sequences. Such sequences were introduced and studied in [2] (and implicitly in [5]) where it was shown that there are boundedly many as well as that they have bounded length. These bounds will be important for us later on.

▶ **Definition 25.** *For a sequence of natural numbers $n_1 \ldots n_t$ its **typical sequence**, $\boldsymbol{\tau(n_1 \ldots n_t)}$, arises from $n_1 \ldots n_t$ by performing the following operations until neither of them is applicable anymore:*

- *Removing consecutive repetitions of entries; or*
- *removing subsequences $n_{i_2} \ldots n_{i_{u-1}}$, with $u \geq 3$ and $\min_{1 \leq j \leq u} n_{i_j} = n_{i_1}$ and $\max_{1 \leq j \leq u} n_{i_j} = n_{i_u}$ or vice versa.*

*A sequence of natural numbers is **typical** if it is the typical sequence of some sequence.*

Note that for a sequence $n_1 \ \dots \ n_t$, its typical sequence can be computed by going through the sequence repeatedly and performing the two operations until no longer possible. As the operations necessarily shorten the sequence, this can be done in time in $\mathcal{O}(t^2)$.

▶ **Lemma 26** ([2, Lemma 3.5]). *There are no more than $\frac{8}{3}2^{2k}$ different typical sequences whose entries are bounded by k.*

▶ **Lemma 27** ([2, Lemma 3.3(ii)]). *A typical sequence whose entries are bounded by k has length at most $2k-1$.*

By iterating Lemma 21 and Lemma 23 and making the additional observation that we never shift vertices into a position of φ whose cutwidth is removed in the typical sequences of the cutwidths between distinguished vertces, we can refine Lemma 24 to:

▶ **Lemma 28.** *Let $1 \leq i \leq s$ and φ be a linear arrangement of H_i. Then there is a linear arrangement ψ' of H that extends φ such that $\mathrm{cw}(\psi') = \min_{\substack{\psi \text{ extension of} \\ \varphi \text{ to } H}} \mathrm{cw}(\psi)$, and for $q \geq 1$ and a position $p \in \{1, \dots, |V(H_i)| - 1 - q\}$ such that $\varphi^{-1}(p)$ and $\varphi^{-1}(p+q+1)$ are distinguished by φ, all $\varphi^{-1}(p+j)$ are not distinguished by φ, no vertex is inserted into any position p' of φ such that $\mathrm{cw}(\varphi, p')$ is removed in $\tau(\mathrm{cw}(\varphi, p) \ \dots \ \mathrm{cw}(\varphi, p+q))$ by ψ'.*

▶ **Definition 29.** *For a linear arrangement φ of H_i with $1 \leq i \leq s$, $p \in \{1, \dots, |V(H_i)| - 1\}$ is a **typical insertion position** of φ, if it is a position into which an extension of φ to H as described in Lemma 28 inserts vertices. 0 and $|V_i|$ are always typical insertion positions of φ. I.e. p is a typical insertion position of φ if p is not removed in $\tau(\mathrm{cw}(\varphi, \max_{\substack{p' \leq p \wedge \varphi^{-1}(p') \text{ is} \\ \text{distinguished by } \varphi}} p') \ \dots \ \mathrm{cw}(\varphi, \min_{\substack{p' > p \wedge \varphi^{-1}(p') \text{ is} \\ \text{distinguished by } \varphi}} p'))$. We define **insertion-typical** linear arrangements inductively: The empty linear arrangement of H_0 is insertion-typical. for $1 \leq i+1 \leq s$, a linear arrangement is a insertion-typical linear arrangement of H_{i+1} if it arises from an insertion-typical linear arrangement of H_i by insertion of the at most one introduced vertex into a typical insertion position of this linear arrangement.*

▶ Remark 30. Let $1 \leq i \leq s$ and φ be a linear arrangement of H_i. Because, by definition, there is always a maximum value entry of $n_1 \ \dots \ n_t$ in $\tau(n_1 \ \dots \ n_t)$, there is a typical insertion position $p \in \{1, \dots, |V(H_i)| - 1\}$ such that $\mathrm{cw}(\varphi) = \mathrm{cw}(\varphi, p)$.

▶ Remark 31. An insertion-typical linear arrangement with cutsidth at most k has at most $(2k-1)^2 \in \mathcal{O}(k^2)$ typical insertion positions. This follows from the fact that at most $2k$ vertices are distinguished by a linear arrangement between every two consecutive of which, by Lemma 27, the cutwidths at typical insertion positions form a sequence of length $\leq 2k-1$.

▶ Remark 32. Let $1 \leq i \leq s$ and φ be an insertion-typical linear arrangement of H_i with a typical insertion position p of φ. Then $p' = \max\{q \in \{0, \dots, |V(H_{i-1})| \mid \varphi(\varphi|^{-1}_{H_{i-1}}(q)) \leq p\}$ is a typical insertion position of $\varphi|_{H_{i-1}}$. Otherwise, assume $\mathrm{cw}(\varphi|_{H_{i-1}}, p')$ to be removed in the typical sequence of the cutwidths among which the cutwidth at p' is considered. Because vertices in $V(H_i) \setminus V(H_{i-1})$ are only inserted into typical insertion positions of $\varphi|_{H_{i-1}}$ by φ, and hyperedges that are considered in H_i but not in H_{i-1} only contain vertices that are distinguished by $\varphi|_{H_{i-1}}$, the cutwidth increase is uniform for the cutwidths among which the cutwidth at p' is considered when moving from $\varphi|_{H_{i-1}}$ to φ. This contradicts p being a typical insertion position of φ.

▶ **Theorem 33.** *If $\mathrm{cw}(H) \leq k$, then there is an insertion-typical linear arrangement ψ of H such that $\mathrm{cw}(\varphi) \leq k$.*

Proof. Let ψ witness $\mathrm{cw}(H) \leq k$. $\psi|_{H_0}$ is trivially insertion-typical. By applying Lemma 28 to $\psi|_{H_0}$ and ψ we can modify ψ in a way that $\psi|_{H_1}$ is insertion-typical while maintaining the same restriction to H_0 and the cutwidth bound of k. (This first step is somewhat pathological, since no actual modification is needed.)
Similarly we can use Lemma 28 on the restriction of this modified ψ to H_1 and this modified ψ to obtain a ψ with an insertion-typical restriction to H_2 while maintaining the same restriction to H_1, and hence also to H_0, and the cutwidth bound of k. Iterating this procedure up to s proved the statement. ◀

4.2 The Dynamic Programming Procedure

Theorem 33 allows us to consider only insertion-typical linear arrangements along the path decomposition. In the following, we describe a dynamic program to construct linear arrangements of the trimmed subhypergraphs induced by the extra nice path decomposition while enforcing a bound of k on their cutwidth. As is usual for algorithms processing path decompositions, we first define our records (i.e. the information stored about each partial solution), what they look like for the initial (empty) stage and how to infer a complete solution from the record at the last stage. Then we describe how to update these records when encountering introduce- and forget bags respectively.

The records. For $1 \leq i \leq s$ our record consists of the following for each insertion-typical linear arrangement φ of H_i with cutwidth at most k:

- The **working information**, which is the weak order $<_\varphi$ on $\{u \in V(H) \mid v_u \in X_i\} \cup \{(e, \mathrm{left}), (e, \mathrm{right}) \mid e \in E(H) \wedge v_e \in X_i\}$ and the cutwidths of φ at each typical insertion position, that is given in the following way:
  - for $v, w \in \{u \in V(H) \mid v_u \in X_i\}$, $v <_\varphi w$, iff $\varphi(v) < \varphi(w)$;
  - for $v \in \{u \in V(H) \mid v_u \in X_i\}$, $v <_\varphi (e, \mathrm{left})$, iff $\varphi(v) < \min_{w \in e \cap V(H_i)} \varphi(w)$ (analogously for (e, right) and $\max_{w \in e \cap V(H_i)} \varphi(w)$);
  - for $v \in \{u \in V(H) \mid v_u \in X_i\}$ and a typical insertion position $p \in \{1, \ldots, |V(H_i)| - 1\}$, $v <_\varphi \mathrm{cw}(\varphi, p)$, iff $\varphi(v) < p$;
  - for (e, left) and a typical insertion position $p \in \{1, \ldots, |V(H_i)| - 1\}$, $v <_\varphi \mathrm{cw}(\varphi, p)$, iff $\min_{w \in e \cap V(H_i)} \varphi(w) < p$ (analogously for (e, right) and $\max_{w \in e \cap V(H_i)} \varphi(w)$); and
  - for typical insertion positions p and q, $\mathrm{cw}(\varphi, p) <_\varphi \mathrm{cw}(\varphi, q)$, iff $p < q$.
- The **solution information**, which in turn consists of
  a map $pos : (\mathrm{cw}(\varphi, p))_{p \text{ a typical insertion position of } \varphi} \to \{0, \ldots, |V(H_i)|\}$, that associates each cutwidth value which is stored in the working information to the typical insertion position at which it is attained;
  a pointer *prec* to the entry in the record at the preceding stage, that corresponds to $\varphi|_{H_{i-1}}$;
  and, if X_i introduces v_u for some $u \in V(H)$, the solution information also specifies $ins \in \{0, \ldots, |V(H_{i-1})|\}$ to be $\mathrm{cw}(\varphi|_{H_{i-1}}, p)$ where p is the typical insertion position of $\varphi|_{H_{i-1}}$ which u is inserted into by φ.

Then the initial record (for $i = 0$) is given by a single cutwidth value 0 as working information, and the single cutwidth value 0 in the working information is mapped to position 1 by *pos*. If the record at stage s is empty then there is no insertion-typical linear arrangement of $H_s = H$ with cutwidth at most k. In this case we output that no linear arrangement of H satisfying cutwidth bound k exists. By Theorem 33 this means $\mathrm{cw}(H) > k$. Otherwise there is an entry in the record at stage s which corresponds to an insertion-typical linear

arrangement of H with (using Remark 30) cutwidth at most k, hence in particular a solution to k-CWLA. We can retrace this linear arrangement using the solution information: Starting the entry in the record at stage i, iteratively traverse the entries corresponding to the restriction at the preceding stage by using the *prec*-pointers in every step. Concurrently save all encountered *ins* in reverse order. By definition, the j-th element of the resulting sequence $\pi = \pi_1 \ \dots \ \pi_{|V(H)|}$ describes the position of the vertex $u \in V(H)$ such that v_u is the j-th of $\{v_w \mid w \in V(H)\}$ that is introduced in our path decomposition, relative to the $j-1$ first such vertices. This allows us to reconstruct the linear arrangement, e.g. as a linked list of vertices of H, representing the order in which it enumerates $V(H)$: For $j = 1, \dots, |V(H)|$ Set the j-th introduced vertex to be between the π_j-th and $(\pi_j + 1)$-th element of the list constructed so far. As inserting into a linked list, takes constant time, the construction runs in time linear in the number of bags of the path decomposition, which in turn is linear in the size of the incidence graph because it is nice.

In the following we describe how to update records when encountering introduce- and forget bags respectively. Proofs for correctness can be given by technical, but straightforward inductions on the stage, using the properties of a nice path decomposition and Remarks 30 and 32. It is also easy to check, using the bound from Remark 31 that the runtime of each of the given procedures lies in $\mathcal{O}(k^3)$.

Introduce bag. Let $1 \leq i \leq s$, assume to have the record for stage i and that X_{i+1} is a bag that introduces $v \in G_I(H)$. We do a case distinction on the type of v:

$v = v_w$ for some $w \in V(H)$. A insertion-typical linear arrangement of H_{i+1} arises from an insertion-typical linear arrangement of H_i by inserting w into a typical insertion position of that linear arrangement. By induction hypothesis, there is an entry in the record at stage i for such a linear arrangement φ which also contains all cutwidths at typical insertion positions of φ in form of the $\mathrm{dom}(<_\varphi) \setminus V(H_i)$. So, for every entry of the record at stage i which corresponds to some φ, and for each element $c \in \mathrm{dom}(<_\varphi) \setminus V(H_i)$, create an entry of the record at stage $i+1$ that corresponds to the linear arrangement φ' arising from φ by inserting w into position $pos(c)$ of φ. The working- and solution information for this entry can be obtained in the following way: Obviously *prec* points to the entry for φ and $ins = pos(c)$.
Define $<_{\varphi'}$ on $\mathrm{dom}(<_\varphi) \setminus \{c\} \cup \{c_{\text{left}}, c_{\text{right}}, w\}$ by setting $x <_{\varphi'} y$ whenever $x, y \in \mathrm{dom}(<_\varphi)$ and $x <_\varphi y$; $x <_{\varphi'} y$ if $x \in \{c_{\text{left}}, c_{\text{right}}, w\}, y \in \mathrm{dom}(<_\varphi)$ and $c <_\varphi y$; and $c_{\text{left}} <_{\varphi'} w <_{\varphi'} c_{\text{right}}$. – Intuitively, in this way we fix the correct relative position of w in φ. The position corresponding to c is split up by the insertion of w into that position.
For every $e \in \{f \in E(H) \mid v_f \in X_{i+1}\} (= \{f \in E(H) \mid v_f \in X_i\})$, if $w \in e$ and $w <_{\varphi'} (e, \text{left})$, increment every cutwidth value d with $w <_{\varphi'} d <_{\varphi'} (e, \text{left})$ by one, and set $(e, \text{left}) <_{\varphi'} x$ iff $w <_{\varphi'} x$(, i.e. '$(e, \text{left}) =_{\varphi'} w$'). Similarly, if $w \in e$ and $(e, \text{right}) <_{\varphi'} w$, we increment every cutwidth value d with $(e, \text{right}) <_{\varphi'} d <_{\varphi'} w$ by one , and set $(e, \text{right}) <_{\varphi'} x$ iff $w <_{\varphi'} x$(, i.e. '$(e, \text{right}) =_{\varphi'} w$'). – Intuitively, in this way we increment the cutwidth values at positions at which a hyperedge contributes to the cutwidth value only after taking w into account. If, at any point, we surpass k by these incrementations, we abort and do not add the entry to the record, as it corresponds to a linear arrangement violating the cutwidth bound.
For every $d \in \mathrm{dom}(<_{\varphi'}) \setminus V(H_{i+1}) \setminus \{c_{\text{left}}, c_{\text{right}}\}$, pos remains the same, if $d <_{\varphi'} c_{\text{left}}$, and is increased by one otherwise. We also set $pos(c_{\text{left}})$ to be the evaluation of the old pos of c and $pos(c_{\text{right}}) = pos(c_{\text{left}}) + 1$. – Intuitively this means we shift the positions in a way to make space for the newly split position, that was created by the insertion of w.
Finally, note that after these modifications our information might include cutwidths at positions that are no longer typical insertion positions (e.g. because the cutwidth values around a position changed). To amend this, we can apply the typical sequence operator τ to

sequences of cutwidth values that are between the remaining elements of $\mathrm{dom}(<_{\varphi'}) \cap (\{u \in V(H) \mid v_u \in X_i\} \cup \{(e, \mathrm{left}), (e, \mathrm{right}) \mid e \in E(H) \wedge v_e \in X_i\})$ and delete the cutwidths removed by this operation from the domains of $<_{\varphi'}$ and *pos*. ◁

$v = v_e$ for some $e \in E(H)$. An insertion-typical linear arrangement of H_{i+1} is by definition also an insertion-typical linear arrangement of H_i. So, for every entry of the record at stage i which corresponds to some φ we create an entry of the record at stage $i+1$ also corresponding to φ. We need to adapt the working- and solution information to take e into account in the cutwidths of φ: Let *prec* point to the entry corresponding to φ in the record at stage i. Because of the extra niceness of our path decomposition there is at least one vertex of e represented in X_{i+1}. Let v be the $<_\varphi$-minimal, and v' to be the $<_\varphi$-maximal $w \in e \cap \{w \in V(H) \mid v_w \in X_{i+1}\}$ ($v = v'$ is possible). Set $(e, \mathrm{left}) <_{\varphi'} x$ iff $v <_{\varphi'} x$(, i.e. '$(e, \mathrm{left}) =_{\varphi'} v$') and set $(e, \mathrm{right}) <_{\varphi'} x$ iff $v' <_{\varphi'} x$(, i.e. '$(e, \mathrm{right}) =_{\varphi'} v'$'). – Intuitively, we find the outermost vertices of e and set the markers (e, left) and (e, right) correspondingly. Leave $<_\varphi$ and *pos* unchanged.
Increment all cutwidth values $c \in \mathrm{dom}(<_{\varphi'})$ with $(e, \mathrm{left}) <_{\varphi'} c <_{\varphi'} (e, \mathrm{right})$ by one. – Intuitively this corresponds to counting e in all cutwidth values, to which it contributes when it is considered. If we surpass k as a cutwidth value by these incrementations, we abort and do not add the entry to the record, as it corresponds to a linear arrangement violating the cutwidth bound. Because this incrementation actually only changes the cutwidth values uniformly for a sequence of cutwidths between distinguished vertices, the typical insertion positions remain unchanged by the consideration of e. ◁

Forget bag. Let $1 \leq i \leq s$, assume to have the record for stage i and that X_{i+1} is a bag that forgets $v \in G_I(H)$. By definition, in this situation, an insertion-typical linear arrangement of H_{i+1} is also an insertion-typical linear arrangement of H_i. Moreover the cutwidth values of any such linear arrangement do not change when moving from H_i to H_{i+1}. Thus, we only have to remove elements from the domain of $<_\varphi$ after forgetting v and make exactly the same amends described in the last paragraph of the case of a bag introducing v_w with $w \in V(H)$, because the removal of certain distinguished vertices might lead to fewer typical insertion positions. For the first step, we remove w from the domain of $<_\varphi$, if $v = v_w$, and remove (e, left) and (e, right) from the domain of $<_\varphi$, if $v = v_e$.

As presented, the dynamic program solves k-CWLA, given a path decomposition of width at most k, in time in $\mathcal{O}(|V(H)| \cdot \mathfrak{b} \cdot k^4)$, where $\mathfrak{b}$ is a bound on the largest number of insertion-typical linear arrangements with cutwidth at most k at a stage. However, at closer inspection, we realise that during the dynamic program solution information is only used to update solution information, and never working information. What is more, solution information is never used to check the cutwidth bound, i.e. to exclude potential linear arrangements, but solely to reconstruct a solution, after a successful traversal of the path decomposition. This implies that actually two insertion-typical linear arrangements with cutwidth at most k can be either both or both not be extended to such a linear arrangement of H if they imply the same solution information. Thus during the algorithm, we only need to add an entry to a record at stage i, if there is no entry with the same solution information already. With this additional argument the runtime of the algorithm lies in $\mathcal{O}(|V(H)| \cdot \mathfrak{w} \cdot k^3)$, where $\mathfrak{w}$ is the number of possibilities for working information of an insertion-typical linear arrangement with cutwidth at most k. We can bound $\mathfrak{w}$ by $(2k)! \cdot (\frac{8}{3} 2^{2k})^{2k-1}$ using Lemma 26 and the bounded width of the path decomposition.

Thus we obtain a linear time algorithm for k-CWLA, by first checking the vertex degree property, in case of a positive answer computing an extra nice tree decomposition and, in case of success applying the described dynamic program, and outputting $\mathrm{cw}(H) > k$ if any stage fails.

▶ **Theorem 34.** *k-CWLA can be solved in time in $\mathcal{O}(|V(H)|)$ and $\mathcal{O}^*(2^{\mathcal{O}(k^2)})$.*

5 Future Work – Using a Tree Decomposition of the Incidence Graph

One can show that the cutwidth of a hypergraph is equal to the product of its maximum vertex degree and the pathwidth of its incidence graph. Hence our algorithm immediately infers an algorithm for solving the Minimum Cut Linear Arrangement Problem in FPT-time parameterised by the maximum vertex degree and the pathwidth of its incidence graph. Reductions in literature show that dropping the restriction on the incidence pathwidth results in NP-hardness [9, Corollary 2.10], and can easily be modified to show the same for dropping the restriction on the maximum vertex degree [9, slight modification of the proof of Lemma 2.4]. (It is not difficult to see that graphs of bounded path- or treewidth also have bounded incidence path- or treewidth respectively.) In this sense, the algorithm is tight.

However the natural question arises, whether incidence pathwidth can be generalised to incidence treewidth. This has been successfully achieved for the analogous algorithm for k-CWLA restricted to graphs [12], and also seems possible for hypergraphs. The idea is, to extend the given dynamic program to also be able to handle tree decompositions of the incidence graph, by specifying the behaviour at *join bags*. In this situation partial linear arrangements for each trimmed subhypergraph corresponding to a bag immediately below the join bag have to "interleave". The number of possibilities for them to do so, is linear in the size of the hypergraph, however we can argue as in Section 4.1 that only interleaving at typical insertion positions of the respective partial linear arrangements needs to be considered.

References

1. Hans L. Bodlaender. A Linear-Time Algorithm for Finding Tree-Decompositions of Small Treewidth. *SIAM Journal on Computing*, 25:1305–1317, December 1996. doi:10.1137/S0097539793251219.
2. Hans L. Bodlaender and Ton Kloks. Efficient and Constructive Algorithms for the Pathwidth and Treewidth of Graphs. *Journal of Algorithms*, 21(2):358–402, 1996. doi:10.1006/jagm.1996.0049.
3. Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Daniel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer Publishing Company, Incorporated, 1st edition, 2015.
4. Jisu Jeong, Eun Jung Kim, and Sang-il Oum. Constructive algorithm for path-width of matroids. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1695–1704, 2016. doi:10.1137/1.9781611974331.ch116.
5. Jens Lagergren and Stefan Arnborg. Finding minimal forbidden minors using a finite congruence. In Javier Leach Albert, Burkhard Monien, and Mario Rodríguez Artalejo, editors, *Automata, Languages and Programming*, pages 532–543, Berlin, Heidelberg, 1991. Springer Berlin Heidelberg.
6. Fillia Makedon and Ivan Hal Sudborough. On Minimizing Width in Linear Layouts. *Discrete Appl. Math.*, 23(3):243–265, June 1989. doi:10.1016/0166-218X(89)90016-4.

7 Fillia S. Makedon, Christos H. Papadimitriou, and Ivan H. Sudborough. Topological bandwidth. In Giorgio Ausiello and Marco Protasi, editors, *CAAP'83*, pages 317–331, Berlin, Heidelberg, 1983. Springer Berlin Heidelberg.

8 Zevi Miller and Ivan Hal Sudborough. A Polynomial Algorithm for Recognizing Bounded Cutwidth in Hypergraphs. *Mathematical Systems Theory*, 24(1):11–40, 1991. doi:10.1007/BF02090388.

9 Burkhard Monien and Ivan Hal Sudborough. Min Cut is NP-Complete for Edge Weighted Treees. *Theor. Comput. Sci.*, 58:209–229, 1988.

10 Neil Robertson and P.D. Seymour. Graph minors. I. Excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983. doi:10.1016/0095-8956(83)90079-5.

11 Dimitrios M. Thilikos, Maria Serna, and Hans L. Bodlaender. Cutwidth I: A linear time fixed parameter algorithm. *Journal of Algorithms*, 56(1):1–24, 2005. doi:10.1016/j.jalgor.2004.12.001.

12 Dimitrios M. Thilikos, Maria Serna, and Hans L. Bodlaender. Cutwidth II: Algorithms for partial w-trees of bounded degree. *Journal of Algorithms*, 56(1):25–49, 2005. doi:10.1016/j.jalgor.2004.12.003.

13 René van Bevern. *Fixed-parameter linear-time algorithms for NP-hard graph and hypergraph problems arising in industrial applications*. PhD thesis, Berlin Institute of Technology, 2014. URL: http://d-nb.info/1058974750.

14 René van Bevern, Rodney G. Downey, Michael R. Fellows, Serge Gaspers, and Frances A. Rosamond. Myhill-Nerode Methods for Hypergraphs. *Algorithmica*, 73(4):696–729, December 2015. doi:10.1007/s00453-015-9977-x.

15 Dong Wang, Edmund M. Clarke, Yunshan Zhu, and James H. Kukula. Using cutwidth to improve symbolic simulation and Boolean satisfiability. In *Proceedings of the Sixth IEEE International High-Level Design Validation and Test Workshop 2001, Monterey, California, USA, November 7-9, 2001*, pages 165–170, 2001. doi:10.1109/HLDVT.2001.972824.