



**HAL**  
open science

## Point-Width and Max-CSPs

Clement Carbonnel, Miguel Romero, Stanislav Živný

► **To cite this version:**

Clement Carbonnel, Miguel Romero, Stanislav Živný. Point-Width and Max-CSPs. ACM Transactions on Algorithms, Association for Computing Machinery, 2020, 16 (4), pp.1-28. 10.1145/3409447 . hal-03088008

**HAL Id: hal-03088008**

**<https://hal.archives-ouvertes.fr/hal-03088008>**

Submitted on 24 Dec 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Point-width and Max-CSPs\*

Clément Carbonnel  
CNRS, University of Montpellier, France  
clement.carbonnel@lirmm.fr

Miguel Romero  
University of Oxford, UK  
miguel.romero@cs.ox.ac.uk

Stanislav Živný  
University of Oxford, UK  
standa.zivny@cs.ox.ac.uk

## Abstract

The complexity of (unbounded-arity) Max-CSPs under structural restrictions is poorly understood. The two most general hypergraph properties known to ensure tractability of Max-CSPs,  $\beta$ -acyclicity and bounded (incidence) MIM-width, are incomparable and lead to very different algorithms.

We introduce the framework of point decompositions for hypergraphs and use it to derive a new sufficient condition for the tractability of (structurally restricted) Max-CSPs, which generalises both bounded MIM-width and  $\beta$ -acyclicity. On the way, we give a new characterisation of bounded MIM-width and discuss other hypergraph properties which are relevant to the complexity of Max-CSPs, such as  $\beta$ -hypertreewidth.

## 1 Introduction

The *Constraint Satisfaction Problem* (CSP) is a well-known framework for expressing a wide range of both theoretical and real-life combinatorial problems [30, 27, 15]. Some examples are satisfiability [35], evaluation of conjunctive queries [10, 28], graph colourings [25] and homomorphisms [26]. An instance of the CSP is a set of *variables*, a domain of *values* and a set of *constraints*; each constraint is a relation applied to a subset of the variables called the constraint scope. Given a CSP instance, the goal is to decide whether one can assign a value to each variable so that all constraints are satisfied; that is, whether for every constraint, the assignment restricted to the constraint scope belongs to the constraint relation. Due to its expressivity, it is not surprising that the CSP is NP-complete in general. This has motivated a long line of research aiming to find tractable restrictions of the problem, sometimes called *islands of tractability*. The focus of this paper is on the so-called *structural restrictions*, which restricts the ways in which the constraints overlap and intersect each other.

---

\*An extended abstract of this work appeared in the *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'19)* [9]. Stanislav Živný was supported by a Royal Society University Research Fellowship. This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 714532). The paper reflects only the authors' views and not the views of the ERC or the European Commission. The European Union is not liable for any use that may be made of the information contained therein. Work done while Clément Carbonnel was at the University of Oxford.

A standard way of analysing structural restrictions is via the *underlying hypergraph* of a CSP instance. The vertex set of this hypergraph is the set of variables  $X$  of the instance and the edges correspond to the scopes of the constraints: each constraint whose scope is a subset  $S \subseteq X$  yields the edge  $S$ . Given a class  $\mathcal{H}$  of hypergraphs, we define the problem  $\text{CSP}(\mathcal{H}, -)$  as the restriction of the CSP to instances whose underlying hypergraphs lie in  $\mathcal{H}$ . Then the goal is to understand for which classes  $\mathcal{H}$  the problem  $\text{CSP}(\mathcal{H}, -)$  is tractable, and for which classes  $\mathcal{H}$  it is not.

The situation of CSP instances of *bounded arity* (i.e., the maximum edge size in the class  $\mathcal{H}$  is a constant) is by now well-understood. In this setting, it follows from [18] and [22] (see also [24]) that  $\text{CSP}(\mathcal{H}, -)$  is tractable if and only if  $\mathcal{H}$  has *bounded treewidth* (under the complexity theoretical assumption that  $\text{FPT} \neq \text{W}[1]$ ). On the other hand, the case of *unbounded arity*, that is, arbitrary classes  $\mathcal{H}$  of hypergraphs, is more delicate. Unlike the bounded-arity case, the complexity of the problem heavily depends on how the constraints in a CSP instance are represented [11]. We focus on one of the most natural and well-studied representation of constraints, namely the *positive* representation, where each constraint is represented by the list of tuples satisfying the constraint.

Bounded treewidth is not the right answer for tractability in the case of unbounded arity, as one can easily find classes  $\mathcal{H}$  of hypergraphs of unbounded treewidth such that  $\text{CSP}(\mathcal{H}, -)$  is tractable. One of the first such classes are the *acyclic* hypergraphs [2, 3, 39] (also called  *$\alpha$ -acyclic* [14]). This tractability result has been extended to more general classes such as hypergraphs of bounded *hypertreewidth* [20] and bounded *fractional hypertreewidth* [23]. The latter is the most general natural hypergraph property known to be tractable, although the precise borderline of polynomial-time solvability is still unknown (and cannot coincide with bounded fractional hypertreewidth; see [29] for a brief discussion on that topic). However, as shown in [29], the classes  $\mathcal{H}$  for which  $\text{CSP}(\mathcal{H}, -)$  becomes *fixed-parameter tractable* (parameterised by the size of the hypergraph) are precisely those of bounded *submodular width*, which are more general than classes of hypergraphs of bounded fractional hypertreewidth.

In this paper we study the problem  $\text{Max-CSP}^1$ , which is a well-known generalisation of CSPs for expressing optimisation problems. Now each constraint is of the form  $f(\mathbf{x})$ , where  $|\mathbf{x}| = r$  and  $f$  is an  $r$ -ary (finite-valued) function  $f : D^r \rightarrow \mathbb{Q}_{\geq 0}$  (we assume that  $f$  is given as the set of pairs  $\{(\mathbf{d}, f(\mathbf{d})) : \mathbf{d} \in D^r, f(\mathbf{d}) > 0\}$ , which corresponds to the positive representation). Given a set of variables  $X = \{x_1, \dots, x_n\}$ , a domain  $D$  of values and a set  $\mathcal{C}$  of (finite-valued) constraints, the goal is to compute the maximum value of  $f(x_1, \dots, x_n) = \sum_{f_c(\mathbf{x}) \in \mathcal{C}} f_c(\mathbf{x})$ , over all possible assignments of values to  $X$ .

In the case of bounded arity, tractability of  $\text{Max-CSP}(\mathcal{H}, -)$  is also characterised by bounded treewidth, which follows directly from the CSP case. However, the complexity of unbounded-arity  $\text{Max-CSP}$ s under structural restrictions is poorly understood and the techniques used in the CSP context cannot be easily applied. Indeed,  $\text{Max-CSP}(\mathcal{H}, -)$  is hard even for classes  $\mathcal{H}$  of  $\alpha$ -acyclic hypergraphs [19]. Moreover, unlike the CSP case, there is no *known* maximal hypergraph property that leads to tractability. The two most general hypergraph properties known to ensure tractability of  $\text{Max-CSP}(\mathcal{H}, -)$  are  *$\beta$ -acyclicity*<sup>2</sup> [4],

<sup>1</sup>A usual definition of a  $\text{Max-CSP}$  instance is a CSP instance with the goal to maximise the number of satisfied constraints. As we explain in Section 2.2, we actually consider a more general framework, sometimes called *finite-valued CSPs* [37] or  $\text{Max-CSP}$ s with payoff functions [31]. Since our main result is a tractability result, this makes it only stronger.

<sup>2</sup>In fact, the authors in [4] consider a more general framework called the *CSP with default values*, and focus on counting solutions. However, they briefly discuss how to adapt the results to the maximisation version.

introduced in [14], and having *bounded (incidence) MIM-width*<sup>3</sup> [38, 34]. These properties are incomparable [4] and lead to very different algorithms. The main goal of this paper is to provide a common explanation for these two tractable properties, and in particular, for all known tractable hypergraph properties for Max-CSPs. We believe that such a unified explanation is a necessary first step to a better understanding of the tractable structural restrictions of Max-CSPs, and ultimately, to a precise characterisation of the tractability frontier.

## 1.1 Contributions

As our main contribution, we introduce the notions of *point decomposition* and *point-width* that unify  $\beta$ -acyclicity and bounded MIM-width. We show that Max-CSPs (with positive representation) are tractable for hypergraphs of bounded point-width, provided a point decomposition of polynomial size and bounded width is also part of the input (Theorem 13). Our tractability result explains the tractability of  $\beta$ -acyclic and bounded MIM-width hypergraphs. In particular, we prove that every  $\beta$ -acyclic hypergraph has a point decomposition of width 1 and polynomial size (Theorem 18), which can be computed in polynomial time. In the case of MIM-width, we obtain a stronger result that may be of independent interest: having bounded MIM-width is equivalent to having bounded *flat point-width* (Theorem 21), where the latter is defined via a syntactic restriction of point decompositions. Finally, we also discuss some related notions such as  $\beta$ -hypertreewidth [21] (Section 7).

The high-level idea behind our new notion of width is that a point decomposition of width  $k \geq 1$  for a hypergraph  $H$  provides a mechanism to encode *several* tree decompositions of hypertreewidth at most  $k$  in a compact and controlled way. In particular, a point decomposition will be expressive enough to encode one such a tree decomposition for *each* subhypergraph of  $H$ . Interestingly, the underlying trees of all these tree decompositions can be very different from each other, as long as they respect the “template” tree  $T$  given by the point decomposition. For *flat* point decompositions, which capture MIM-width, these underlying trees need to be *subtrees* of the template  $T$ , and then they are more similar to each other. The full details of point decompositions and their flat variant are given in Sections 3 and 6, respectively.

The algorithm behind our main tractability result (Theorem 13) uses a form of dynamic programming over the point decomposition where in each step we need to solve an instance of the *weighted maximum independent set* problem in *chordal graphs* (which is known to be tractable and in fact solvable in linear time [17], see also [?]). We can think of this procedure as doing dynamic programming *simultaneously* over all the tree decompositions of the subhypergraphs of  $H$  encoded in the point decomposition.

## 1.2 Related work

It is also possible to parameterise CSPs and Max-CSPs by a class of admissible underlying *structures*, instead of hypergraphs, which offers a more fine-grained analysis. In the case of CSPs of bounded arity, a complete classification of the tractable cases in terms of the

---

<sup>3</sup>The results for MIM-width in [38, 34] apply to Max-SAT (and #SAT), but can be adapted to Max-CSPs. Let us also remark that in [38, 34] a more general notion than that of bounded MIM-width, namely having polynomial *PS-width*, is shown to be tractable for Max-SAT and #SAT. This notion is however not purely structural, as it depends on the entire input instance and not just its hypergraph.

underlying relational structures follows from [12] and [22]. Recently, a similar classification has been obtained for (finite-valued) Max-CSPs in terms of the underlying *valued* structures [8].

Another important type of restrictions (and perhaps the most studied one) are the *non-uniform* restrictions, where the constraint relations (or functions) are restricted to be fixed. In this case, the situation is fairly clear and now, after two decades of intense research, complete classifications have been obtained for CSPs [5, 40], and (finite-valued) Max-CSPs [37].

### 1.3 Structure

The paper is organised as follows. Section 2 introduces the necessary notation on hypergraphs and Max-CSPs. Section 3 defines point decompositions and point-width. The main tractability result is given in Section 4. Sections 5 and 6 show that  $\beta$ -acyclicity and bounded MIM-width are special cases of bounded point-width, respectively. We conclude in Section 7.

## 2 Preliminaries

### 2.1 Hypergraphs, points and covers

We assume that the reader is familiar with elementary graph theory and refer to Diestel's textbook for more details [13]. Given a graph  $G$ , we use  $V(G)$  and  $E(G)$  to denote its sets of *vertices* and *edges*, respectively. The subgraph of a graph  $G$  *induced* by a set  $X \subseteq V(G)$ , denoted by  $G[X]$ , has vertex set  $X$  and edge set  $\{\{u, v\} \in E(G) : u, v \in X\}$ . We use the same notation for directed graphs.

**Hypergraphs.** A (finite) hypergraph is a finite set of non-empty finite sets called *edges*. The set of *vertices* of a hypergraph  $H$ , denoted by  $V(H)$ , is the union of all its edges. Note that in this definition, every vertex of a hypergraph belongs to at least one edge. A *subhypergraph* of a hypergraph  $H$  is a subset of  $H$ . We use  $S(H)$  to denote the set of all vertex sets of subhypergraphs of  $H$ .

**Points.** A *point* of a hypergraph  $H$  is a pair  $(v, e)$  with  $e \in H$  and  $v \in e$ . We use  $P(H)$  to denote the set of all points of  $H$ . Given  $P \subseteq P(H)$  and  $e \in H$ , the *restriction of  $e$  to  $P$* , denoted by  $e|_P$ , is the set  $\{v \in e : (v, e) \in P\}$ . By extension the *restriction of  $H$  to  $P$* , denoted by  $H|_P$ , is the hypergraph  $\{e|_P : e \in H, e|_P \neq \emptyset\}$ . If  $H'$  is a subhypergraph of  $H$  and  $P \subseteq P(H)$ , we use the notation  $H'|_P$  as a shorthand for  $H'|_{P \cap P(H')}$ .

**Remark 1.** The reader might find the following duality, observed by an anonymous reviewer, useful (although we will not use it). Hypergraphs  $H$  are in bijection with their incidence graphs  $inc(H)$ . (Incidence graphs are defined in Section 6.) The set  $P(H)$  of points is then the edge set of  $inc(H)$ . The restriction  $H|_P$  is a subhypergraph of  $H$  induced by the edge set  $P$  in  $inc(H)$ . The sets  $S(H)$  are the open neighborhoods of the subsets of  $E(H)$  in  $inc(H)$ .

**Covers.** An *edge cover* of a hypergraph  $H$  is a subhypergraph  $C$  of  $H$  such that  $V(C) = V(H)$ . The *cover number* of  $H$ , denoted by  $cn(H)$ , is the smallest cardinality of an edge cover of  $H$ . We denote by  $\beta\text{-}cn(H)$  the maximum of  $cn(H')$  over all subhypergraphs  $H'$  of  $H$ .

### 2.2 Max-CSP

A finite-valued *function* of arity  $r = \text{ar}(f)$  over a domain  $D$  is a mapping  $f : D^r \rightarrow \mathbb{Q}_{\geq 0}$ . A finite-valued *constraint* over a set  $X$  of variables is an expression of the form  $f(\mathbf{x})$ , where  $f$  is

a finite-valued function and  $\mathbf{x} \in X^{\text{ar}(f)}$ . The set of variables appearing in  $\mathbf{x}$  is called the *scope* of the constraint  $f(\mathbf{x})$ . An instance  $I$  of the Max-CSP problem is a finite set  $X = \{x_1, \dots, x_n\}$  of variables, a finite domain  $D$  of values, and an objective function of the form

$$f_I(x_1, \dots, x_n) = \sum_{i=1}^q f_i(\mathbf{x}_i)$$

where each  $f_i(\mathbf{x}_i)$ ,  $1 \leq i \leq q$  is a finite-valued constraint. The goal is to compute the maximum value of  $f_I$  over all possible assignments to  $X$ , which we denote by  $\text{opt}(I)$ . In this paper we assume that each function  $f_i$ ,  $1 \leq i \leq q$  is given in the input as the table of all pairs  $(\mathbf{d}, f_i(\mathbf{d}))$  where  $\mathbf{d} \in D^{\text{ar}(f_i)}$  and  $f_i(\mathbf{d}) > 0$  (the so-called *positive representation*). It follows that the total size  $\|I\|$  of a Max-CSP instance  $I$  is roughly

$$\sum_{i=1}^q \left( \text{ar}(f_i) \log(|X|) + \sum_{\substack{\mathbf{d} \in D^{\text{ar}(f_i)} \\ f_i(\mathbf{d}) > 0}} (\text{ar}(f_i) \log(|D|) + |\text{enc}(f_i(\mathbf{d}))|) \right)$$

where  $\text{enc}(\cdot)$  is a reasonable encoding for rational numbers.

Actually, Max-CSPs are commonly defined with only  $\{0, 1\}$ -valued functions, or with  $\{0, w\}$ -valued functions, where  $w$  could be different in different functions; the latter are called weighted Max-CSPs. What we defined as Max-CSPs is a more general framework, sometimes called *finite-valued CSPs* [37] or Max-CSPs with payoff functions [31].

The hypergraph of a Max-CSP instance is the set of scopes of its constraints. Given a family  $\mathcal{H}$  of hypergraphs, we denote by  $\text{Max-CSP}(\mathcal{H}, -)$  the restriction of Max-CSP to the instances whose hypergraph belongs to  $\mathcal{H}$ .

Without loss of generality, we will always assume that no two constraints share the same scope and for every constraint  $f_i(\mathbf{x}_i)$ , the entries of  $\mathbf{x}_i$  are pairwise distinct. Given a Max-CSP instance  $I$  with hypergraph  $H$  and  $e \in H$ , we will use  $f_e(\mathbf{x}_e)$  to denote the unique constraint with scope  $e$ . Given a constraint  $f_e(\mathbf{x}_e)$  with  $e \in H$ , its *support* is the relation  $R_e := \{\mathbf{d} \in D^{|e|} : f_e(\mathbf{d}) > 0\}$ . Without ambiguity we will sometimes treat  $R_e$  as a set of assignments to  $e$ . If  $\psi : X' \rightarrow D$  is an assignment to  $X' \subseteq X$ , we define  $\text{val}(\psi) = \sum_{e \in H: e \subseteq X'} f_e(\psi(\mathbf{x}_e))$  and call  $\psi$  a *partial assignment to  $X$* . In particular, for any partial assignment  $\psi$  to  $X$ , we have that  $\text{val}(\psi) \leq \text{opt}(I)$ . Finally, given a partial assignment  $\psi : X' \rightarrow D$ , we say that  $\psi$  *satisfies* an edge  $e \in H$  if  $\psi|_{X' \cap e} \in R_e|_{X' \cap e}$ , and satisfies a subhypergraph if it satisfies all of its edges. Note that  $\psi$  can satisfy edges that are not completely contained in  $X'$ .

### 3 Point decompositions and point-width

Let  $H$  be a hypergraph. Let  $\mathcal{T} = (T, (B_t)_{t \in V(T)})$  be a pair such that  $T$  is a rooted tree and  $B_t \subseteq P(H)$  is a set of points, for every  $t \in V(T)$ . For  $t \in V(T)$ , we call the set  $B_t$  the *bag of  $t$*  and the pairs  $(t, S)$  with  $S \in S(H|_{B_t})$  the *sub-bags of  $t$* . We denote by  $<_T$  the strict partial order on  $V(T)$  such that  $t_1 <_T t_2$  if and only if  $t_1$  is a descendant of  $t_2$  in  $T$ . A  $\mathcal{T}$ -*structure* is a directed graph  $A$  whose vertex set is the set of all sub-bags of  $V(T)$  and such that for every arc  $((t_1, S_1), (t_2, S_2))$  in  $A$  we have  $t_1 <_T t_2$ .

**Example 1.** Consider the hypergraph  $H = \{e, e_1, e_2, e_3\}$ , where  $e = \{x_0, x_1, x_2, x_3\}$  and  $e_i = \{x_0, x_i\}$ , for every  $i \in \{1, 2, 3\}$ ; see Figure 1 on the left. In particular,  $V(H) = \{x_0, x_1, x_2, x_3\}$ .

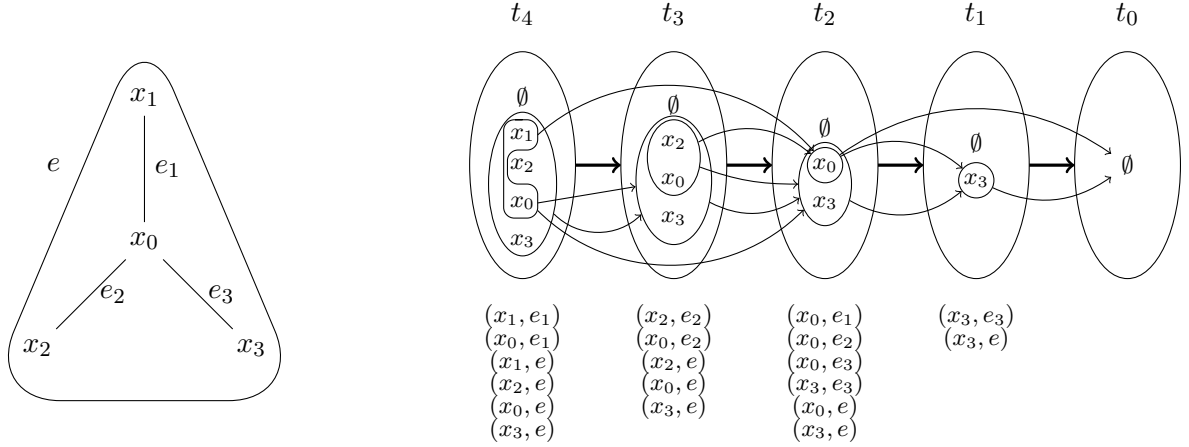


Figure 1: The hypergraph  $H$  and its point decomposition from Examples 1–6.

The right-hand side of Figure 1 depicts a pair  $\mathcal{T} = (T, (B_t)_{t \in V(T)})$ , where  $T$  is a path (depicted by bold arcs<sup>4</sup>) rooted at  $t_0$ , and the points in each bag  $B_t$  are listed below each node. The sub-bags of each node of  $T$  are depicted within the node. For instance, for the node  $t_4$  we have  $H|_{B_{t_4}} = \{\{x_1, x_0\}, \{x_1, x_2, x_0, x_3\}\}$ . Hence the sub-bags of  $t_4$  are  $(t_4, \emptyset)$ ,  $(t_4, \{x_1, x_0\})$  and  $(t_4, \{x_1, x_2, x_0, x_3\})$ . The arcs between sub-bags represent a possible  $\mathcal{T}$ -structure  $A$ .

**Definition 2** (Decomposability). *Let  $A$  be a  $\mathcal{T}$ -structure for a pair  $\mathcal{T} = (T, (B_t)_{t \in V(T)})$ . We say that  $A$  is decomposable if for any two arcs  $(s_1, s)$ ,  $(s_2, s)$  in  $A$ , if*

- (i)  $s_1, s_2$  are sub-bags of different vertices of  $V(T)$ , and
- (ii) there exist two sub-bags  $s'_1, s'_2$  (not necessarily distinct) of the same vertex  $t \in V(T)$ , and directed paths in  $A$  from  $s'_1$  to  $s_1$ , and from  $s'_2$  to  $s_2$

then either  $(s_1, s_2) \in E(A)$  or  $(s_2, s_1) \in E(A)$ .

Observe that if  $A$  is *not* decomposable due to arcs  $(s_1, s)$ ,  $(s_2, s)$ , where  $s_1, s_2$  are sub-bags of  $t_1, t_2 \in V(T)$ , respectively, then either  $t_1 <_T t_2$  or  $t_2 <_T t_1$  must hold (otherwise, condition (ii) would fail). Let say that  $t_1 <_T t_2$ . Note that it could be possible that  $t = t_1$ , in which case, the directed path from  $s'_1$  to  $s_1$  is simply the empty path, i.e.,  $s'_1 = s_1$ . If additionally,  $s'_2 = s_1$ , we obtain the simplest case of non-decomposability, in which there is a directed path in  $A$  from  $s_1$  to  $s_2$  (and  $(s_1, s_2) \notin E(A)$ ).

**Example 2.** The  $\mathcal{T}$ -structure  $A$  from Example 1 and Figure 1 is decomposable. Consider for instance the arcs  $(s_1, s)$  and  $(s_2, s)$  with  $s = (t_2, \{x_0, x_3\})$ ,  $s_1 = (t_4, \{x_1, x_0\})$  and  $s_2 = (t_3, \{x_2, x_0, x_3\})$ . We have that  $s_1$  and  $s_2$  are sub-bags of different vertices of  $T$ , and condition (ii) of decomposability holds if we take  $s'_1 = s_1$  and  $s'_2 = (t_4, \{x_0, x_1, x_2, x_3\})$ . In this case decomposability requires that at least one of  $(s_1, s_2)$  or  $(s_2, s_1)$  is an arc of  $A$ , which is true for  $(s_1, s_2)$ .

<sup>4</sup>We view the tree  $T$  as undirected although there is an implicit direction by the parent/child relationship. For clarity, in Figure 1, we directed the (bold) edges of the tree  $T$  towards the root.

The intuition behind decomposability is as follows. Suppose we have a sub-bag  $s$  in the  $\mathcal{T}$ -structure and two incoming arcs  $(s_1, s), (s_2, s)$  in  $A$ , where  $s_1, s_2$  are sub-bags of distinct vertices  $t_1, t_2 \in V(T)$ . Let  $T_{s_1}$  be the rooted subtree of  $T$  induced by all the nodes in  $V(T)$  that can “reach”  $s_1$ , i.e., that contain a sub-bag  $s'_1$  from which  $s_1$  is reachable in  $A$ . Similarly, we define  $T_{s_2}$ . Note that the root of  $T_{s_1}$  and  $T_{s_2}$  is  $t_1$  and  $t_2$ , respectively. Then decomposability means that whenever  $s_1$  and  $s_2$  are “incomparable” with respect to  $A$  (i.e., neither  $(s_1, s_2)$  nor  $(s_2, s_1)$  is an arc), then  $T_{s_1}$  and  $T_{s_2}$  must be disjoint subtrees.

**Definition 3** (Realisations). *Let  $A$  be a  $\mathcal{T}$ -structure for a pair  $\mathcal{T} = (T, (B_t)_{t \in V(T)})$ . A realisation of  $A$  is a subgraph  $A'$  of  $A$  induced by a subset  $X \subseteq V(A)$  such that*

- (i)  $X$  contains at most one sub-bag of each  $t \in V(T)$ , and
- (ii)  $A'$  has exactly one sink, which must be a sub-bag of the root of  $T$ .

For any realisation  $A'$  of a  $\mathcal{T}$ -structure  $A$ , we define  $T_{A'}$  as the rooted tree whose vertex set is

$$V(T_{A'}) = \{t \in V(T) : \exists \text{ a sub-bag } (t, S) \in V(A')\},$$

and whose edges are defined as follows. Suppose  $t_1, t_2 \in V(T_{A'})$  due to sub-bags  $(t_1, S_1), (t_2, S_2) \in V(A')$ , respectively. Then  $t_2$  is the parent of  $t_1$ , i.e.,  $(t_1, t_2) \in E(T_{A'})$ , if  $t_2$  is the least vertex with respect to  $<_T$  of the set

$$\{t \in V(T) : \exists (t, S) \in V(A') \text{ and } ((t_1, S_1), (t, S)) \in E(A')\}.$$

**Example 3.** For the  $\mathcal{T}$ -structure  $A$  in Figure 1, consider the subgraph  $A_1$  of  $A$  induced by the sub-bags  $(t_4, \{x_1, x_0\}), (t_3, \{x_2, x_0, x_3\}), (t_2, \{x_0, x_3\}), (t_1, \{x_3\})$  and  $(t_0, \emptyset)$ . We have that  $A_1$  is a realisation as the only sink is  $(t_0, \emptyset)$ . Note that if we remove from  $A_1$  the sub-bag  $(t_1, \{x_3\})$  then we obtain a subgraph that is not a realisation as now  $(t_2, \{x_0, x_3\})$  becomes a sink. Observe also that  $T_{A_1}$  is precisely  $T$ . Another possible realisation is the subgraph  $A_2$  of  $A$  induced by the sub-bags  $(t_4, \{x_1, x_0\}), (t_3, \{x_2, x_0\}), (t_2, \{x_0\})$  and  $(t_0, \emptyset)$ . In this case,  $T_{A_2}$  is the tree with vertices  $\{t_0, t_2, t_3, t_4\}$  and edges  $(t_2, t_0), (t_3, t_2)$  and  $(t_4, t_2)$ .

For a  $\mathcal{T}$ -structure  $A$  and a subhypergraph  $H'$  of  $H$ , we denote by  $A[H']$  the subgraph of  $A$  induced by the set  $\{(t, V(H'|_{B_t})) : t \in V(T)\}$ . We denote by  $A[H']_\emptyset$  the directed graph obtained from  $A[H']$  after removing every connected component  $C$  in  $A[H']$  that satisfies the following: for every sub-bag  $(t, S) \in C$ , we have that  $t$  is not the root of  $T$  and  $S = \emptyset$ . In other words,  $A[H']_\emptyset$  contains precisely the connected components of  $A[H']$  that contain a sub-bag of the root of  $T$  or a sub-bag  $(t, S)$  with  $S \neq \emptyset$ .

**Example 4.** The subgraph  $A_2$  of  $A$  from Example 3 is precisely  $A[H']_\emptyset$ , where  $H' = \{e_1, e_2\}$ . Note that  $(t_1, \emptyset)$  needs to be removed from  $A[H']$  in order to obtain  $A[H']_\emptyset$ . While  $A[H']_\emptyset$  is a realisation,  $A[H']$  is not, as  $(t_1, \emptyset)$  is a sink.

**Definition 4** (Point decomposition). *A point decomposition of a hypergraph  $H$  is a triple  $(T, (B_t)_{t \in V(T)}, A)$  where  $T$  is a rooted tree, each set  $B_t \subseteq P(H)$  is a set of points of  $H$ ,  $A$  is a decomposable  $\mathcal{T}$ -structure, where  $\mathcal{T} = (T, (B_t)_{t \in V(T)})$ , and*

- (i) For every edge  $e \in H$ , there exists  $t \in V(T)$  such that  $P(\{e\}) = \{(v, e) : v \in e\} \subseteq B_t$ .
- (ii) For every subhypergraph  $H'$  of  $H$ , the subgraph  $A[H']_\emptyset$  of  $A$  is a realisation.



(iii) For every realisation  $A'$  of  $A$  and  $v \in \cup_{(t,S) \in V(A')} S$ , the set

$$\{t \in V(T_{A'}) : \exists (t, S) \in V(A') \text{ and } v \in S\}$$

induces a connected subtree of  $T_{A'}$ .

A point decomposition is *flat* if every arc in  $A$  is between sub-bags of nodes adjacent in  $T$ . The *width* of a point decomposition  $(T, (B_t)_{t \in V(T)}, A)$  of a hypergraph  $H$  is given by  $\max_{t \in V(T)} \beta\text{-cn}(H|_{B_t})$ , the *point-width* of  $H$ , denoted by  $\text{pw}(H)$ , is the minimum width over all its point decompositions, and the *flat point-width* of  $H$ , denoted by  $\text{fpw}(H)$ , is the minimum width over all its flat point decompositions.

Throughout the paper we assume a straightforward encoding for point decompositions, where each bag is given as a list of points, the tree  $T$  is given as a rooted graph whose vertex set is the set of all bags, and the  $\mathcal{T}$ -structure  $A$  is given as a directed graph whose vertex set is the set of all sub-bags. We denote by  $\|P\|$  the encoding size of a point decomposition  $P$ . We remark that checking whether a triple  $(T, (B_t)_{t \in V(T)}, A)$  is a point decomposition may be a difficult task due to conditions (ii) and (iii). Whether it can be done in polynomial time is an interesting question, which we leave for future work.

**Example 5.** Figure 1 shows a point decomposition of the hypergraph  $H$  to the left. Note that  $\beta\text{-cn}(H|_{B_{t_i}}) = 1$ , for  $1 \leq i \leq 4$ , and then the width of the decomposition is 1. Hence  $\text{pw}(H) = 1$ . Note that the decomposition is not flat.

As mentioned in the introduction, the intuition is that a  $\mathcal{T}$ -structure  $A$  in a point decomposition of width  $k$  encodes various tree decompositions of hypertreewidth at most  $k$  (cf. Appendix A for a precise definition of tree decomposition and hypertreewidth), and in particular, one for each subhypergraph  $H'$  of  $H$ . Such a tree decomposition for  $H'$  is given by the tree  $T_{A[H']_\emptyset}$  and the bags correspond to the sub-bags in  $A[H']_\emptyset$ .

Finally, let us remark that once we know the  $\mathcal{T}$ -structure of a point decomposition, the particular form of the tree  $T$  is irrelevant. Indeed, we can always assume that  $T$  is a path: if it is not the case, we can extend  $<_T$  to a total order  $<_{\text{tot}}$  on  $V(T)$ , which is precisely  $<_{T'}$  for a certain path  $T'$ , and then replace  $T$  by  $T'$  in the point decomposition. However, in the case of *flat* point decompositions this is not true. Hence, in general, we shall not impose any assumption on the tree  $T$ .

## 4 The algorithm

In this section we describe a polynomial-time algorithm for solving Max-CSPs when the input instance is paired with a point decomposition of bounded width of its hypergraph. We start with a number of simple definitions and observations before proving the main result in Theorem 13.

**Definition 5** (Partial realisations). *Let  $H$  be a hypergraph and  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ . A partial realisation of  $A$  is a subgraph  $A'$  of  $A$  induced by a subset  $X \subseteq V(A)$  such that (i)  $X$  contains at most one sub-bag of each  $t \in V(T)$ , (ii)  $A'$  has exactly one sink  $s$  and (iii) there is a (possibly empty) directed path in  $A$  from  $s$  to a sub-bag of the root of  $T$ .*

The rooted tree  $T_{A'}$  of a partial realisation  $A'$  is defined the same way as for realisations: its vertex set is the set of all  $t \in V(T)$  with at least one sub-bag in  $V(A')$ , and the parent of  $t_1 \in V(T_{A'})$  with  $(t_1, S_1) \in V(A')$  is the least vertex with respect to  $<_T$  in the set  $\{t \in V(T) : \exists(t, S) \in V(A') \text{ and } ((t_1, S_1), (t, S)) \in E(A')\}$ . The next observation is a minor extension of condition (iii) of point decompositions to partial realisations.

**Observation 6.** *Let  $H$  be a hypergraph,  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ ,  $A'$  be a partial realisation of  $A$  and  $v \in \cup_{(t, S) \in V(A')} S$ . Then, the set*

$$\{t \in V(T_{A'}) : \exists(t, S) \in V(A') \text{ and } v \in S\}$$

*induces a connected subtree of  $T_{A'}$ .*

*Proof.* Let  $s$  be the unique sink of  $A'$ . If  $s$  is a sub-bag of the root of  $T$  then  $A'$  is a realisation and the claim follows from condition (iii) of point decompositions. Otherwise, let  $(s, s_1, \dots, s_n)$  be a directed path in  $A$  from  $s$  to a sub-bag  $s_n$  of the root of  $T$ . The subgraph  $A^*$  of  $A$  induced by  $V(A') \cup \{s_1, \dots, s_n\}$  is a realisation and  $T_{A'}$  is precisely the subtree of  $T_{A^*}$  rooted at  $s$ , so the observation follows.  $\square$

**Definition 7 (Guards).** *Let  $H$  be a hypergraph,  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$  and  $(t, S)$  be a sub-bag of  $t \in V(T)$ . A guard of  $(t, S)$  is an inclusion-minimal subhypergraph  $H'$  of  $H$  such that  $V(H'|_{B_t}) = S$ .*

**Definition 8 (Consistent assignments).** *Let  $H$  be the hypergraph of a Max-CSP instance and  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ . If  $s = (t, S)$  is a sub-bag of  $t \in V(T)$ , an  $s$ -valid assignment is an assignment  $\psi : S \rightarrow D$  such that  $\psi$  satisfies some guard  $C$  of  $s$ . A consistent assignment to a partial realisation  $A'$  of  $A$  is a function  $\phi$  that maps every sub-bag  $s = (t, S) \in V(A')$  to an  $s$ -valid assignment such that for any two sub-bags  $(t_1, S_1), (t_2, S_2)$  with  $t_1, t_2$  adjacent in  $T_{A'}$ ,  $\phi((t_1, S_1))|_{S_1 \cap S_2} = \phi((t_2, S_2))|_{S_1 \cap S_2}$ .*

The following is a direct consequence from Observation 6.

**Observation 9.** *Let  $H$  be the hypergraph of a Max-CSP instance,  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ ,  $\phi$  be a consistent assignment to some partial realisation  $A'$  of  $A$  and  $X' := \cup_{(t, S) \in V(A')} S$ . Then, there exists an assignment  $\psi : X' \rightarrow D$  such that for every  $s = (t, S) \in V(A')$ ,  $\phi(s) = \psi|_S$ .*

**Definition 10.** *Let  $H$  be the hypergraph of a Max-CSP instance,  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ ,  $\phi$  be a consistent assignment to a partial realisation  $A'$  of  $A$  and  $\psi$  be as in Observation 9. The value of  $(\phi, A')$  is the quantity*

$$\text{val}(\phi, A') := \sum_{e \in H : \exists(t, S) \in V(A'), e \subseteq S} f_e(\psi(\mathbf{x}_e)).$$

The general idea behind the algorithm is to traverse the tree  $T$  of the point decomposition bottom-up, keeping track for each sub-bag  $s$  and  $s$ -valid assignment  $\psi$  of the best value achievable by a partial realisation  $A'$  with sink  $s$  and consistent assignment to  $A'$  that agrees with  $\psi$  on  $s$ . The fact that  $A$  is decomposable ensures that joining multiple partial realisations to a common sink always produces a partial realisation, as long as their initial sinks form an independent set in a certain (easily computable) chordal graph. This property enables a dynamic programming approach. It will follow from conditions (i), (ii) and (iii) in the definition of point decompositions that the maximum of the values computed by this algorithm at the root of  $T$  is, in fact, the optimum of the Max-CSP instance.

**Proposition 11.** *Let  $I$  be a Max-CSP instance with hypergraph  $H$  and  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ . The maximum of  $\text{val}(\phi, A')$  over all realisations  $A'$  of  $A$  and consistent assignments  $\phi$  to  $A'$  is exactly  $\text{opt}(I)$ .*

*Proof.* Let  $M$  be the maximum of  $\text{val}(\phi, A')$  over all realisations  $A'$  of  $A$  and consistent assignments  $\phi$  to  $A'$ .

We first prove  $M \geq \text{opt}(I)$ . Let  $\psi_{\text{opt}}$  be an assignment to the variables of  $I$  such that  $\text{val}(\psi_{\text{opt}}) = \text{opt}(I)$ , and let  $H' \subseteq H$  be the set of edges satisfied by  $\psi_{\text{opt}}$ . Consider the subgraph  $A[H']_{\emptyset}$  of  $A$ , which by condition (ii) of point decompositions is a realisation. We define  $\phi^*$  as the function that maps each  $(t, S) \in V(A[H']_{\emptyset})$  to  $\psi_{\text{opt}}|_S$ . Since  $\psi_{\text{opt}}$  satisfies  $H'$ , it satisfies at least one guard for each sub-bag  $(t, S) \in V(A[H']_{\emptyset})$ . Therefore,  $\phi^*$  is a consistent assignment to  $A[H']_{\emptyset}$ . By condition (i) of point decompositions, for every edge  $e \in H'$  there exists  $(t, S) \in V(A[H']_{\emptyset})$  such that  $e \subseteq S$ , and hence  $M \geq \text{val}(\phi^*, A[H']_{\emptyset}) = \text{opt}(I)$ .

We now prove  $\text{opt}(I) \geq M$ . Let  $A'$  be a realisation of  $A$  and  $\phi$  be a consistent assignment to  $A'$  such that  $\text{val}(\phi, A') = M$ . By Observation 9, there exists an assignment  $\psi$  to  $X' := \cup_{(t,S) \in V(A')} S$  such that

$$\text{val}(\psi) = \sum_{e \in H: e \subseteq X'} f_e(\psi(\mathbf{x}_e)) \geq \sum_{e \in H: \exists (t,S) \in V(A'), e \subseteq S} f_e(\psi(\mathbf{x}_e)) = \text{val}(\phi, A') = M$$

and hence  $\text{opt}(I) \geq M$ . □

If  $A'$  is a partial realisation and  $s \in V(A')$ , we use  $A'[s]$  to denote the partial realisation induced by the sub-bags  $s'$  of  $A'$  such that there is a (possibly empty) directed path in  $A'$  from  $s'$  to  $s$ .

**Observation 12.** *Let  $H$  be the hypergraph of a Max-CSP instance,  $(T, (B_t)_{t \in V(T)}, A)$  be a point decomposition of  $H$ ,  $\phi$  be a consistent assignment to a partial realisation  $A'$  of  $A$  with sink  $s = (t, S)$  and  $\psi$  be as in Observation 9. Let  $W$  be the set of all sub-bags  $s' = (t', S')$  in  $V(A')$  such that  $t'$  is a child of  $t$  in  $T_{A'}$ . Then,*

$$\text{val}(\phi, A') = \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{\substack{s' \in W \\ s' = (t', S')}} \left( \text{val}(\phi|_{V(A'[s'])}, A'[s']) - \sum_{e \in H: e \subseteq S \cap S'} f_e(\psi(\mathbf{x}_e)) \right).$$

*Proof.* By definition of  $T_{A'}$  there is no arc  $(s_1, s_2)$  in  $A$  with  $s_1, s_2 \in W$ . Since  $A$  is decomposable, it follows that the sets  $V(A'[s'])$ ,  $s' \in W$ , are pairwise disjoint. Furthermore, by Observation 6, if there exist an edge  $e \in H$  and two sub-bags  $s_1, s_2 \in W$  with  $e \subseteq (\cup_{(t^*, S^*) \in V(A'[s_1])} S^*) \cap (\cup_{(t^*, S^*) \in V(A'[s_2])} S^*)$  then  $e \subseteq S$ . Similarly, if there exist  $e \in H$  and  $s_1 = (t_1, S_1) \in W$  such that  $e \subseteq (\cup_{(t^*, S^*) \in V(A'[s_1])} S^*) \cap S$ , then  $e \subseteq S_1$ . Putting everything together we have

$$\begin{aligned}
\text{val}(\phi, A') &= \sum_{e \in H: \exists (t^*, S^*) \in V(A'), e \subseteq S^*} f_e(\psi(\mathbf{x}_e)) \\
&= \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{s' \in W} \left( \sum_{e \in H, e \not\subseteq S: \exists (t^*, S^*) \in V(A'[s']), e \subseteq S^*} f_e(\psi(\mathbf{x}_e)) \right) \\
&= \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{\substack{s' \in W \\ s' = (t', S')}} \left( \text{val}(\phi|_{V(A'[s']), A'[s']}) - \sum_{e \in H: e \subseteq S \cap S'} f_e(\psi(\mathbf{x}_e)) \right)
\end{aligned}$$

as claimed.  $\square$

Recall that an independent set in a graph is a subset of vertices that induces a subgraph with no edges. We will denote by  $\text{IS}(G)$  the set of all independent sets in a graph  $G$ .

**Theorem 13.** *Let  $k$  be a fixed positive integer. There exists an algorithm which, given as input a Max-CSP instance  $I$  with hypergraph  $H$  and a point decomposition  $P = (T, (B_t)_{t \in V(T)}, A)$  of  $H$  of width at most  $k$ , computes  $\text{opt}(I)$  in time polynomial in  $\|P\|$  and  $\|I\|$ .*

*Proof.* We first describe the algorithm. To each bag  $t \in V(T)$ , sub-bag  $s = (t, S)$  and  $s$ -valid assignment  $\psi$  we will associate a rational value  $\text{val}_{\text{alg}}(s, \psi)$ . We will compute these values bottom-up, starting from the leaves of  $T$ .

Let  $t$  be a vertex of  $T$ ,  $s = (t, S)$  be a sub-bag of  $t$  and  $\psi$  be an  $s$ -valid assignment. Suppose that the values  $\text{val}_{\text{alg}}(s', \psi')$  have already been computed for all pairs  $(s' = (t', S'), \psi')$  with  $t' <_T t$ . If  $t$  is a leaf then we set  $\text{val}_{\text{alg}}(s, \psi) := \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e))$ . If  $t$  is not a leaf then we define a vertex-weighted graph  $G$  where

- $V(G)$  is the set of all sub-bags  $s' = (t', S')$  with  $t' <_T t$  such that (i) there exists at least one  $s'$ -valid assignment  $\psi'$  such that  $\psi'|_{S \cap S'} = \psi|_{S \cap S'}$  and (ii)  $(s', s)$  is an arc in  $A$ ;
- $E(G)$  is the set of all pairs  $\{(t_1, S_1), (t_2, S_2)\} \in V(G)^2$  such that either  $t_1 = t_2$  or  $((t_1, S_1), (t_2, S_2))$  is an arc in  $A$ ;
- For every  $s' = (t', S') \in V(G)$ , the weight  $w(s')$  of  $s'$  is the maximum of

$$\text{val}_{\text{alg}}(s', \psi') - \sum_{e \in H: e \subseteq S \cap S'} f_e(\psi(\mathbf{x}_e))$$

over all  $s'$ -valid assignments  $\psi'$  such that  $\psi'|_{S \cap S'} = \psi|_{S \cap S'}$ . Note that this quantity is well-defined because at least one suitable assignment  $\psi'$  exists, by definition of  $V(G)$ .

We then set  $\text{val}_{\text{alg}}(s, \psi) := \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \max_{U \in \text{IS}(G)} (\sum_{s' \in U} w(s'))$ . Once  $\text{val}_{\text{alg}}(s, \psi)$  is computed for all pairs  $(s, \psi)$  where  $s$  is a sub-bag of the root of  $T$ , the algorithm outputs the maximum of  $\text{val}_{\text{alg}}(s, \psi)$  over all such pairs.

**Claim 1.** *For every  $t \in V(T)$ , sub-bag  $s = (t, S)$  with a (possibly empty) directed path in  $A$  from  $s$  to a sub-bag of the root of  $T$  and  $s$ -valid assignment  $\psi$ ,  $\text{val}_{\text{alg}}(s, \psi)$  is the maximum of  $\text{val}(\phi, A')$  over all partial realisations  $A'$  of  $A$  whose sink is  $s$  and consistent assignments  $\phi$  to  $A'$  such that  $\phi(s) = \psi$ .*

*Proof.* We proceed by induction, proving the claim for all pairs  $(s, \psi)$  in the same order the algorithm computes  $\text{val}_{\text{alg}}(s, \psi)$ . Let  $s = (t, S)$  be a sub-bag with a directed path in  $A$  to a sub-bag of the root of  $T$  and  $\psi$  be an  $s$ -valid assignment. Suppose that the claim holds for all pairs  $(s', \psi')$  for which  $\text{val}_{\text{alg}}(s', \psi')$  is computed by the algorithm before  $\text{val}_{\text{alg}}(s, \psi)$  (and in particular for all pairs  $(s', \psi')$  where  $s'$  is a sub-bag of  $t'$  with  $t' <_T t$ ). If  $t$  is a leaf then the claim trivially holds, so suppose that  $t$  is not a leaf. Let  $A'$  be any partial realisation of  $A$  with sink  $s$  and  $\phi$  be a consistent assignment to  $A'$  with  $\phi(s) = \psi$ . Let  $W$  be the set of all sub-bags  $s' = (t', S')$  in  $V(A')$  such that  $t'$  is a child of  $t$  in  $T_{A'}$ . Note that we have  $t' <_T t$  for all  $(t', S')$  in  $W$ ; it follows from the definition of the tree  $T_{A'}$  that there does not exist an arc  $((t', S'), (t'', S''))$  in  $A$  with  $(t', S'), (t'', S'') \in W$  (as otherwise one of  $t', t''$  would not have  $t$  as parent in  $T_{A'}$ ). Therefore,  $W$  is a subset of  $V(G)$  and forms an independent set. By Observation 12 and the induction hypothesis we have

$$\begin{aligned} \text{val}(\phi, A') &= \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{\substack{s' \in W \\ s' = (t', S')}} \left( \text{val}(\phi|_{V(A'[s'])}, A'[s']) - \sum_{\substack{e \in H: \\ e \subseteq S \cap S'}} f_e(\psi(\mathbf{x}_e)) \right) \\ &\leq \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{\substack{s' \in W \\ s' = (t', S')}} \left( \text{val}_{\text{alg}}(s', \phi(s')) - \sum_{\substack{e \in H: \\ e \subseteq S \cap S'}} f_e(\psi(\mathbf{x}_e)) \right). \end{aligned}$$

Then, from the definition of the vertex weights in  $G$  we deduce

$$\text{val}(\phi, A') \leq \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{s' = (t', S') \in W} w(s')$$

and since  $\text{val}_{\text{alg}}(s, \psi)$  is the maximum of the right-hand side expression taken over all independent sets  $W'$  of  $G$ , we finally obtain that  $\text{val}(\phi, A') \leq \text{val}_{\text{alg}}(s, \psi)$ , as claimed.  $\blacksquare$

At this point, we need only prove that there exist a partial realisation  $A'$  with sink  $s$  and a consistent assignment  $\phi$  to  $A'$  such that  $\phi(s) = \psi$  and  $\text{val}(\phi, A')$  is exactly  $\text{val}_{\text{alg}}(s, \psi)$ . Let  $W$  be the independent set of  $G$  chosen by the algorithm to compute  $\text{val}_{\text{alg}}(s, \psi)$ . For each sub-bag  $s' = (t', S') \in W$ , let  $\psi_{s'}$  be an  $s'$ -valid assignment such that  $\text{val}_{\text{alg}}(s', \psi_{s'}) - \sum_{e \in H: e \subseteq S \cap S'} f_e(\psi(\mathbf{x}_e)) = w(s')$  and  $\psi_{s'}|_{S \cap S'} = \psi|_{S \cap S'}$ . Note that every sub-bag in  $W$  can reach a sub-bag of the root of  $T$  via a directed path in  $A$  by going through  $s$ . Then, by induction for each  $s' \in W$  there exist a partial realisation  $A'_{s'}$  with sink  $s'$  and a consistent assignment  $\phi_{s'}$  to  $A'_{s'}$  such that  $\phi_{s'}(s') = \psi_{s'}$  and  $\text{val}(\phi_{s'}, A'_{s'}) = \text{val}_{\text{alg}}(s', \psi_{s'}) = w(s') + \sum_{e \in H: e \subseteq S \cap S'} f_e(\psi(\mathbf{x}_e))$ . Now, if we define  $A'$  as the subgraph of  $A$  induced by  $\{s\} \cup (\cup_{s' \in W} V(A'_{s'}))$ , then (i)  $A'$  has a single sink  $s$ , since the sinks of each  $A'_{s'}$  have an outgoing arc to  $s$ , and (ii)  $A'$  contains at most one sub-bag for each  $t \in V(T)$  because  $A$  is decomposable and  $W$  is an independent set in  $G$ . It follows that  $A'$  is a partial realisation of  $A$ .

The mapping  $\phi$  defined on  $V(A')$  such that  $\phi(s^*) := \psi$  if  $s^* = s$  and  $\phi(s^*) := \phi_{s'}(s^*)$  otherwise, where  $s'$  is the only sub-bag in  $W$  such that  $s^* \in V(A'_{s'})$ , is a consistent assignment

to  $A'$ . Finally, by Observation 12 and the induction hypothesis we obtain

$$\begin{aligned} \text{val}(\phi, A') &= \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{\substack{s' \in W \\ s' = (t', S')}} \left( \text{val}(\phi_{s'}, A'_{s'}) - \sum_{e \in H: e \subseteq S \cap S'} f_e(\psi(\mathbf{x}_e)) \right) \\ &= \sum_{e \in H: e \subseteq S} f_e(\psi(\mathbf{x}_e)) + \sum_{s' \in W} w(s') \end{aligned}$$

which is exactly  $\text{val}_{\text{alg}}(s, \psi)$ .

**Corollary 14.** *The output of the algorithm is the maximum of  $\text{val}(\phi, A')$  over all realisations  $A'$  of  $A$  and consistent assignments  $\phi$  to  $A'$ .*

We deduce from Corollary 14 and Proposition 11 that the algorithm correctly outputs  $\text{opt}(I)$ . We now turn to the problem of estimating the time complexity of the algorithm. To this end, we will need to bound the time necessary to compute the maximum-weight independent sets. This will be achieved with the help of the next claim.

A graph is *chordal* if every cycle  $C$  with at least four vertices has a *chord*, that is, an edge connecting two vertices that are not consecutive in  $C$ .

**Claim 2.** *For any given pair  $(s, \psi)$ , the associated graph  $G$  is chordal.*

*Proof.* By way of contradiction let us assume that there exists a pair  $(s, \psi)$  for which  $G$  has a chordless cycle  $C$ . Let  $s_1 = (t_1, S_1)$  be a sub-bag in  $C$  such that  $t_1$  is minimal with respect to  $<_T$ . Since  $C$  is chordless, at least one of the two sub-bags that are adjacent to  $s_1$  in  $C$  is not a sub-bag of  $t_1$ . Let  $s_2$  be that sub-bag, and  $s_3$  be the other one. Note that  $s_2$  and  $s_3$  are not adjacent in  $G$ , which means that they are not sub-bags of the same vertex of  $T$  and none of  $(s_2, s_3), (s_3, s_2)$  is an arc in  $A$ . Furthermore, since  $t_1$  is minimal with respect to  $<_T$  in the cycle, there is a directed path (of length 1) in  $A$  from  $s_1$  to  $s_2$ . Likewise, there is always a directed path in  $A$  from some sub-bag of  $t_1$  to  $s_3$ : if  $s_3$  is a sub-bag of  $t_1$  then this path is empty, and otherwise we have the path  $(s_1, s_3)$  in  $A$  by minimality of  $t_1$ . Finally, by construction we have the arcs  $(s_2, s)$  and  $(s_3, s)$  in  $A$ , so the triple  $(s, s_2, s_3)$  contradicts the decomposability of  $A$ . Thus the chordless cycle  $C$  does not exist, which establishes the claim.  $\blacksquare$

**Claim 3.** *The runtime of the algorithm is polynomial in  $\|I\|$  and  $\|P\|$ .*

*Proof.* By definition of the width of a point decomposition, for each bag  $B_t$ ,  $t \in V(T)$  we have  $\beta\text{-cn}(H|_{B_t}) \leq k$ . Hence, for each subhypergraph  $H' \subseteq H$  there exists a subhypergraph  $H^* \subseteq H'$ ,  $|H^*| \leq k$ , such that  $V(H^*|_{B_t}) = V(H'|_{B_t})$ ; in particular, every guard of a sub-bag contains at most  $k$  edges. Therefore, given a sub-bag  $s$ , any  $s$ -valid assignment is in the join of the projections of the support of at most  $k$  constraints; it follows that there are at most  $|H|^k q^k$  distinct  $s$ -valid assignments, where  $q := \max_{e \in H} |R_e|$ , and the algorithm computes  $\text{val}_{\text{alg}}(s, \psi)$  for  $O(\|P\| |H|^k q^k)$  pairs  $(s, \psi)$ .

The computation of  $\text{val}_{\text{alg}}(s, \psi)$  for a given pair  $(s, \psi)$  reduces to computing a maximum weighted independent set in the graph  $G$ , which can be achieved in time linear in  $\|G\| = O(\|P\|)$  since  $G$  is chordal [17, ?] by Claim 2. Constructing the graph  $G$  takes time polynomial in  $\|P\|$  and  $|H|^k q^k$ , which concludes the proof of Claim 3.  $\blacksquare$

Theorem 13 now follows from Corollary 14, Proposition 11 and Claim 3.  $\square$

## 5 Relationship with $\beta$ -acyclicity

A hypergraph  $H$  is  $\alpha$ -acyclic [3] if it has a *join tree*. A join tree is a pair  $(T, \lambda)$  where  $T$  is a tree and  $\lambda$  is a bijection from  $V(T)$  to (the edges of)  $H$ , such that for every  $v \in V(H)$  the set  $\{t \in V(T) : v \in \lambda(t)\}$  induces a connected subtree of  $T$ . A hypergraph  $H$  is  $\beta$ -acyclic [14] if every subhypergraph of  $H$  is  $\alpha$ -acyclic. It is known that  $\beta$ -acyclic hypergraphs are tractable for Max-CSPs:

**Theorem 15** ([4]). *Max-CSP( $\mathcal{H}, -$ ) can be solved in polynomial time if  $\mathcal{H}$  is a family of  $\beta$ -acyclic hypergraphs.*

The algorithm of Brault-Baron, Capelli, and Mengel [4] works by variable elimination, making use of a well-known alternative characterisation of  $\beta$ -acyclic hypergraphs in terms of the so-called  $\beta$ -elimination orders [3]. In this section we show that such hypergraphs are covered by our framework as they always have a point decomposition of polynomial size and width 1, which can be computed in polynomial time. Hence, together with Theorem 13, we can obtain Theorem 15.

An ordering  $(x_1, \dots, x_n)$  of the vertices of a hypergraph  $H$  is a  $\beta$ -elimination order if for any  $x_i \in V(H)$  and  $e, e' \in H$  such that  $x_i \in e \cap e'$ , either  $e \cap \{x_j : j \geq i\} \subseteq e'$  or  $e' \cap \{x_j : j \geq i\} \subseteq e$ . A hypergraph is  $\beta$ -acyclic if and only if it has a  $\beta$ -elimination order [3].

Our construction of point decompositions for  $\beta$ -acyclic hypergraphs is inspired by recent work of Capelli [7], from whom we borrow some notation and lemmas. Let  $H$  be a  $\beta$ -acyclic hypergraph and  $<_\beta$  be a  $\beta$ -elimination order of  $H$ . Given a vertex  $x \in V(H)$ , let  $V(H)_{\leq x} := \{v \in V(H) : v \leq_\beta x\}$  and  $V(H)_{\geq x} := \{v \in V(H) : v \geq_\beta x\}$ . Let  $<_H$  be the total order on the edges of  $H$  such that  $e_1 <_H e_2$  if and only if  $\max_{<_\beta}(e_1 \Delta e_2) \in e_2$ , where  $\Delta$  denotes the symmetric difference. A *walk* from  $e \in H$  to  $f \in H$  is a sequence  $(e_1, x_1, e_2, x_2, \dots, x_{n-1}, e_n)$ , with  $n \geq 1$ , where each  $e_i$  is an edge of  $H$ ,  $e_1 = e$ ,  $e_n = f$ , and each  $x_i$  is a vertex of  $H$  such that  $x_i \in e_i \cap e_{i+1}$ . Given  $x \in V(H)$  and  $e \in H$ , let  $H_e^x$  denote the set of edges of  $H$  reachable from  $e$  through a walk that contains only vertices  $\leq_\beta x$  and edges  $\leq_H e$ .

**Example 6.** Consider the hypergraph  $H$  from Figure 1 defined as  $H = \{e, e_1, e_2, e_3\}$ , where  $e = \{x_0, x_1, x_2, x_3\}$  and  $e_i = \{x_0, x_i\}$ , for  $i \in \{1, 2, 3\}$ . We have that  $H$  is  $\beta$ -acyclic. A possible  $\beta$ -elimination order is  $x_1 <_\beta x_2 <_\beta x_0 <_\beta x_3$ . The induced order  $<_H$  is  $e_1 <_H e_2 <_H e_3 <_H e$ . For instance, note that  $e_1 \notin H_{e_3}^{x_2}$  as the only possible walk would be  $(e_3, x_0, e_1)$  but  $x_0 >_\beta x_2$ . We have  $H_{e_3}^{x_2} = \{e_3\}$  and  $H_{e_3}^{x_0} = \{e_3, e_1, e_2\}$ . Note that  $e \notin H_{e_3}^{x_0}$  as  $e >_H e_3$ .

**Lemma 16** ([7, Lemma 2]). *Let  $x, y \in V(H)$  such that  $x \leq_\beta y$  and  $e, f \in H$  such that  $e \leq_H f$  and  $V(H_e^x) \cap V(H_f^y) \cap V(H)_{\leq x} \neq \emptyset$ . Then,  $H_e^x \subseteq H_f^y$ .*

**Theorem 17** ([7, Theorem 3]). *For every  $x \in V(H)$  and  $e \in H$ ,  $V(H_e^x) \cap V(H)_{\geq x} \subseteq e$ .*

Now we are ready to state the main result of this section:

**Theorem 18.** *Every  $\beta$ -acyclic hypergraph has a point decomposition of polynomial size and width 1. Moreover, such a decomposition can be computed in polynomial time.*

*Proof.* Let  $H$  be a  $\beta$ -acyclic hypergraph with  $\beta$ -elimination order  $<_\beta$ . The rooted tree  $T$  of the point decomposition of  $H$  has one vertex  $t_x$  for each vertex  $x \in V(H)$ , plus a special vertex  $t_\perp$ . The root of  $T$  is  $t_\perp$  and its only child is  $t_z$ , where  $z$  is the last vertex in the  $\beta$ -elimination

order of  $H$ . The remainder of  $T$  is then a path, where  $t_x$  is the child of  $t_y$  if and only if  $y$  is the vertex that directly follows  $x$  in the  $\beta$ -elimination order. In particular, for any two vertices  $x, y \in V(H)$  we have that  $t_x <_T t_y$  if and only if  $x <_\beta y$ .

For any  $t_x \in V(T)$ , the associated bag  $B_{t_x}$  is the set of all points  $(y, e) \in P(H)$  with  $x \in e$  and  $x \leq_\beta y$ . The bag of  $t_\perp$  is an empty set of points. We denote by  $\mathcal{T}$  the pair  $(T, (B_t)_{t \in V(T)})$ .

By definition of a  $\beta$ -elimination order, for each  $t_x \in V(T)$  it holds that  $\beta\text{-cn}(H|_{B_{t_x}}) = 1$  and the possible sub-bags are of the form  $(t_x, e \cap V(H)_{\geq x})$  with  $e \in H$ . We now describe the directed graph  $A$  on the sub-bags of  $\mathcal{T}$  that will complete the point decomposition. Given any two sub-bags  $s_x = (t_x, S_x)$  and  $s_y = (t_y, S_y)$  with  $x, y \in V(H)$  and  $x <_\beta y$ , we add an arc from  $s_x$  to  $s_y$  if one of the following conditions is satisfied:

- ( $\dagger$ )  $|S_x| = 1$  and there exist  $e, f \in H$  such that  $S_x = e \cap V(H)_{\geq x}$ ,  $S_y = f \cap V(H)_{\geq y}$  and  $e \in H_f^y$ ;
- ( $\dagger\dagger$ )  $|S_x| > 1$  and there exist  $e, f \in H$  such that  $S_x = e \cap V(H)_{\geq x}$ ,  $S_y = f \cap V(H)_{\geq y}$ ,  $e \in H_f^y$  and  $y \leq_\beta z$ , where  $z = \min_{<_\beta}(S_x \setminus \{x\})$ .

In addition, if  $|S_x| = 1$  we add the arc  $((t_x, S_x), (t_\perp, \emptyset))$ . Figure 1 shows the construction applied to the  $\beta$ -acyclic hypergraph  $H$  to the left and  $\beta$ -elimination order  $x_1 <_\beta x_2 <_\beta x_0 <_\beta x_3$ .

By construction,  $A$  is a  $\mathcal{T}$ -structure. The next claim will be used in conjunction with Lemma 16 and Theorem 17 to show that  $A$  is decomposable.

**Claim 4.** *Let  $s_x = (t_x, S_x)$  and  $s_y = (t_y, S_y)$  be two sub-bags with  $x, y \in V(H)$  and  $S_x, S_y \neq \emptyset$ , such that there is a directed path in  $A$  from  $s_x$  to  $s_y$ . Then, there exist  $e, f \in H$  such that  $S_x = e \cap V(H)_{\geq x}$ ,  $S_y = f \cap V(H)_{\geq y}$  and  $e \in H_f^y$ .*

*Proof.* We prove the claim by induction on the length of the path. If the path has length 1 (i.e.  $(s_x, s_y)$  is an arc in  $A$ ) then  $(s_x, s_y)$  satisfies either ( $\dagger$ ) or ( $\dagger\dagger$ ) and the claim holds. Now, suppose that the path has length  $n > 1$  and that the claim holds for all paths of length  $n - 1$ . Let  $z \in V(H)$ ,  $z <_\beta y$ , be such that  $s_z = (t_z, S_z)$  is the predecessor of  $s_y$  in the path. (Note that such a vertex  $z$  always exists because the special sub-bag  $(t_\perp, \emptyset)$  is a sink in  $A$ .) By induction, there exist  $e_x, f_z \in H$  such that  $S_x = e_x \cap V(H)_{\geq x}$ ,  $S_z = f_z \cap V(H)_{\geq z}$  and  $e_x \in H_{f_z}^z$ . Also, since  $(s_z, s_y)$  is an arc in  $A$ , it satisfies either ( $\dagger$ ) or ( $\dagger\dagger$ ) and hence there exist  $e_z, f_y \in H$  such that  $S_z = e_z \cap V(H)_{\geq z}$ ,  $S_y = f_y \cap V(H)_{\geq y}$  and  $e_z \in H_{f_y}^y$ . In particular, there exists a walk  $w_{f_z e_x}$  from  $f_z$  to  $e_x$  that only contains vertices  $\leq_\beta z$  and edges  $\leq_H f_z$ , and a walk  $w_{f_y e_z}$  from  $f_y$  to  $e_z$  that only contains vertices  $\leq_\beta y$  and edges  $\leq_H f_y$ .

If  $f_z <_H f_y$ , then  $(w_{f_y e_z}, z, w_{f_z e_x})$  is a walk from  $f_y$  to  $e_x$  that contains only vertices  $\leq_\beta y$  and edges  $\leq_H f_y$ . Therefore, we have  $e_x \in H_{f_y}^y$  and the claim follows from the edges  $e_x, f_y$ . If instead we have  $f_y <_H f_z$ , then by Theorem 17 we have  $f_z \cap V(H)_{\geq y} = e_z \cap V(H)_{\geq y} \subseteq V(H_{f_y}^y) \cap V(H)_{\geq y} \subseteq f_y$ . Note that  $f_z \cap V(H)_{\geq y}$  cannot be a strict subset of  $f_y \cap V(H)_{\geq y}$  because  $f_y <_H f_z$ . This implies that  $f_z \cap V(H)_{\geq y} = f_y \cap V(H)_{\geq y} = S_y$ . Finally, we deduce from the inclusion  $H_{f_z}^z \subseteq H_{f_z}^y$  that  $e_x \in H_{f_z}^y$ , and the claim follows from the edges  $e_x, f_z$ . ■

**Claim 5.**  *$A$  is decomposable.*

*Proof.* We prove the claim by contradiction. Suppose that  $A$  is not decomposable, that is, there exist five sub-bags  $s, s_x = (t_x, S_x), s_y = (t_y, S_y), s_z^1 = (t_z, S_z^1), s_z^2 = (t_z, S_z^2)$  with  $x, y, z \in V(H)$  and  $x \neq y$  such that (i)  $(s_x, s)$  and  $(s_y, s)$  are arcs in  $A$ , (ii) neither  $(s_x, s_y)$  nor  $(s_y, s_x)$  is



an arc in  $A$ , and (iii) there are directed paths in  $A$  from  $s_z^1$  to  $s_x$  and from  $s_z^2$  to  $s_y$ . By the definition of  $A$ , we can further assume that none of  $S_x, S_y, S_z^1, S_z^2$  is empty.

By Claim 4, there exist  $f_x, e_z^1, f_y, e_z^2 \in H$  such that  $S_x = f_x \cap V(H)_{\geq x}$ ,  $S_y = f_y \cap V(H)_{\geq y}$ ,  $S_z^1 = e_z^1 \cap V(H)_{\geq z}$ ,  $S_z^2 = e_z^2 \cap V(H)_{\geq z}$ ,  $e_z^1 \in H_{f_x}^x$  and  $e_z^2 \in H_{f_y}^y$ . Without loss of generality we assume  $x <_\beta y$ .

We distinguish two cases:

- $f_x \leq_H f_y$ . Observe that  $z \in e_z^1 \cap e_z^2 \cap V(H)_{\leq x} \subseteq V(H_{f_x}^x) \cap V(H_{f_y}^y) \cap V(H)_{\leq x}$ , so by Lemma 16 we have  $H_{f_x}^x \subseteq H_{f_y}^y$ . In particular, it holds that  $f_x \in H_{f_y}^y$ . Since  $(s_x, s_y)$  is not an arc in  $A$ , we can deduce that  $|S_x| > 1$ ; it follows that  $s$  is of the form  $(t_w, S_w)$  where  $w \leq_\beta \min_{<_\beta}(S_x \setminus \{x\})$ . However, the arc  $(s_y, s)$  implies that  $y <_\beta w$ , which means that  $(s_x, s_y)$  should have been an arc in  $A$ , a contradiction.
- $f_x \geq_H f_y$ . Then, we have  $z \in V(H_{f_x}^y) \cap V(H_{f_y}^y) \cap V(H)_{\leq y}$ , so by Lemma 16 we have  $H_{f_y}^y \subseteq H_{f_x}^y$ . By Theorem 17 it holds that  $f_y \cap V(H)_{\geq y} \subseteq f_x$ , and in particular  $y \in f_x$ . Then, since  $(s_x, s)$  is an arc in  $A$  and  $|S_x| = |f_x \cap V(H)_{\geq x}| > 1$  (as it contains both  $x$  and  $y$ ), it follows that  $s$  is of the form  $(t_w, S_w)$  where  $w \leq_\beta \min_{<_\beta}(S_x \setminus \{x\})$ . Again, the arc  $(s_y, s)$  implies that  $y <_\beta w$ . Finally, since  $y \in S_x \setminus \{x\}$ , we have  $w \leq_\beta \min_{<_\beta}(S_x \setminus \{x\}) \leq_\beta y <_\beta w$ , a contradiction.

■

**Claim 6.** *The triple  $(T, (B_t)_{t \in V(T)}, A)$  is a point decomposition of  $H$ .*

*Proof.*  $T$  is a rooted tree, each  $B_t$  with  $t \in V(T)$  is a set of points, and  $A$  is a decomposable  $\mathcal{T}$ -structure by Claim 5. That leaves conditions (i), (ii) and (iii) in the definition of a point decomposition to verify.

By construction, for any edge  $e \in H$ , we have that  $P(\{e\}) = \{(v, e) : v \in e\} \subseteq B_{t_x}$ , where  $x \in V(H)$  is the smallest vertex in  $e$  with respect to  $<_\beta$ . Hence condition (i) holds.

For condition (ii), let  $H'$  be a subhypergraph of  $H$ . Note that  $A' := A[H']_\emptyset$  is precisely the subgraph of  $A$  induced by

$$\{(t_\perp, \emptyset)\} \cup \{(t_x, V(H'|_{B_{t_x}})) : x \in V(H), V(H'|_{B_{t_x}}) \neq \emptyset\}.$$

because all sub-bags of the form  $(t_x, \emptyset)$  with  $x \in V(H)$  are isolated sub-bags of non-root vertices of  $T$ . We show that  $A'$  is a realisation of  $A$ . Suppose for the sake of contradiction that it is not the case. The only possibility is that  $A'$  has two sinks, and one of them is of the form  $s_x = (t_x, S_x)$  with  $x \in V(H)$  and  $S_x \neq \emptyset$ . The sub-bag  $s_\perp = (t_\perp, \emptyset)$  belongs to  $V(A')$ , which implies that  $|S_x| > 1$  since otherwise  $(s_x, s_\perp)$  would be an arc in  $A'$  and hence  $s_x$  would not be a sink. Now, let  $y = \min_{<_\beta}(S_x \setminus \{x\})$ , and let  $e_x \in H'$  be such that  $S_x = e_x \cap V(H)_{\geq x}$ . Let  $s_y = (t_y, S_y)$  denote the sub-bag  $(t_y, V(H'|_{B_{t_y}}))$  and  $e_y \in H'$  be such that  $S_y = e_y \cap V(H)_{\geq y}$ . Note that  $S_y$  is not empty because  $(y, e_x) \in B_{t_y}$ ; this implies in particular that  $s_y \in V(A')$ . If  $e_x \cap V(H)_{\geq y} = e_y \cap V(H)_{\geq y}$  then  $(s_x, s_y)$  would be an arc in  $A$  because of condition (††) (with  $(e, f) = (e_x, e_x)$ ). Since  $s_y \in V(A')$ , this contradicts our hypothesis that  $s_x$  is a sink in  $A'$ . On the other hand, if  $e_x \cap V(H)_{\geq y} \neq e_y \cap V(H)_{\geq y}$  then from the facts that  $<_\beta$  is a  $\beta$ -elimination order,  $e_x \in H'$  and  $y \in e_x$ , we can further assume that  $e_x \cap V(H)_{\geq y} \subset e_y \cap V(H)_{\geq y}$ . It follows that  $e_x <_H e_y$ , and the walk  $(e_y, y, e_x)$  implies that  $e_x \in H_{e_y}^y$ . However, by condition (††) we deduce that  $(s_x, s_y)$  is an arc in  $A$ , a final contradiction.

For condition (iii), we first prove that for any arc  $(s, s')$  of  $A'$  where  $s = (t_y, S_y)$ ,  $y \in V(H)$  and  $s' = (t', S')$  it holds that  $S_y \setminus S' = \{y\}$ . Observe that  $S_y$  always contains  $y$ , and  $S'$  may only contain vertices  $z \in V(H)$  with  $y <_H z$ , so  $S_y \setminus S' = \{y\}$  whenever  $(s, s')$  satisfies condition  $(\dagger)$  or if  $s' = (t_\perp, \emptyset)$ . If  $(s, s')$  satisfies condition  $(\dagger\dagger)$  instead, then  $s' = (t_z, S')$  for some  $z \leq_\beta \min_{<_\beta}(S_y \setminus \{y\})$ . Let  $e_y, f_z \in H$  be such that  $S_y = e_y \cap V(H)_{\geq y}$ ,  $S' = f_z \cap V(H)_{\geq z}$  and  $e_y \in H_{f_z}^z$ . By Theorem 17 we have that  $S_y \setminus \{y\} = e_y \cap V(H)_{\geq z} \subseteq V(H_{f_z}^z) \cap V(H)_{\geq z} \subseteq f_z$  and hence  $S_y \setminus S' = S_y \setminus (f_z \cap V(H)_{\geq z}) = \{y\}$ , as claimed.

Now, let  $A'$  be a realisation of  $A$  and  $x \in \cup_{(t,S) \in V(A')} S$ . It follows from the property above that if  $t'$  is the parent of  $t$  in  $T_{A'}$  and  $(t, S), (t', S')$  are the sub-bags in  $V(A')$ , then  $x \in S$  and  $x \notin S'$  if and only if  $t = t_x$ . Since  $x$  may only appear in a set  $S_y$  for sub-bags of the form  $(t_y, S_y)$  with  $y \leq_\beta x$ , the set

$$\{t \in V(T_{A'}) : \exists (t, S) \in V(A') \text{ and } x \in S\}$$

induces a connected subtree of  $T_{A'}$ , which proves the claim.  $\blacksquare$

The point decomposition  $(T, (B_t)_{t \in V(T)}, A)$  has polynomial size. Moreover, it can be computed in polynomial time since a  $\beta$ -elimination order can be computed efficiently from  $H$  [4]. Recall that for each  $t_x \in V(T)$  it holds that  $\beta\text{-cn}(H|_{B_{t_x}}) = 1$ ; it follows that  $(T, (B_t)_{t \in V(T)}, A)$  has width 1. Together with Claim 6, these last observations establish Theorem 18.  $\square$

In the case of the hypergraph  $H$  of Figure 1, it can be verified that our construction produces a non-flat point decomposition independently of the  $\beta$ -elimination order we pick for  $H$ . As we shall see in the next section, this is not coincidence as  $\beta$ -acyclic hypergraphs cannot be captured by flat point decompositions of *any* constant width. The reason is that the latter captures precisely the so-called hypergraphs of constant MIM-width, which are known to be incomparable with  $\beta$ -acyclic hypergraphs [4].

## 6 Flat point-width and MIM-width

In this section, we show how our main tractability result from Theorem 13 also explains the tractability of Max-CSPs for classes of hypergraphs of bounded MIM-width [38, 34]. Before doing so, we need some notation and definitions.

An *induced matching* in a graph  $G$  is a set  $M \subseteq E(G)$  such that no two edges of  $M$  share a common vertex and for every edge  $e = \{u, v\} \in E(G) \setminus M$ , we have  $\{u, v\} \not\subseteq \cup_{\{u', v'\} \in M} \{u', v'\}$ . For a graph  $G$ , we denote by  $\text{MIM}(G)$  the maximum size of an induced matching in  $G$ . A graph  $G$  is *bipartite* if there is a partition  $V_1, V_2$  of its vertex set  $V(G)$  such that every edge of  $G$  has one endpoint in  $V_1$  and the other in  $V_2$ . For a graph  $G$  and disjoint subsets  $V_1, V_2$  of  $V(G)$ , we define  $G[V_1, V_2]$  to be the bipartite graph with vertex set  $V_1 \cup V_2$  that contains all edges of  $G$  with one endpoint in  $V_1$  and the other in  $V_2$ .

A *branch decomposition* of a graph  $G$  is a pair  $(T, \delta)$  where  $T$  is a binary rooted tree and  $\delta$  is a bijection from  $V(G)$  to the leaves of  $T$ . For  $t \in V(T)$ , we let  $T_t$  denote the subtree of  $T$  rooted at  $t$  and  $V_t$  denote the set  $\{\delta^{-1}(\ell) : \ell \text{ is a leaf of } T_t\}$ . The *MIM-width* of the branch decomposition  $(T, \delta)$  is the maximum  $\text{MIM}(G[V_t, V(G) \setminus V_t])$ , taken over all  $t \in V(T)$ . The MIM-width [38] of  $G$ , denoted by  $\text{mimw}(G)$ , is the minimum MIM-width over all branch decompositions of  $G$ .

The *incidence graph* of a hypergraph  $H$ , denoted by  $\text{inc}(H)$ , is the bipartite graph with vertex set  $V(H) \cup H$  and edge set  $\{\{v, e\} : v \in V(H), e \in H \text{ and } v \in e\}$ . We define the

MIM-width  $\text{mimw}(H)$  of the hypergraph  $H$  to be  $\text{mimw}(\text{inc}(H))$ . It follows from the work of Sæther, Telle and Vatshelle [34] that Max-CSPs are tractable for hypergraphs of bounded MIM-width, provided a branch decomposition of bounded MIM-width is given with the input. More formally:

**Theorem 19** ([34]). *Let  $k \geq 1$  be fixed. There exists an algorithm which, given as input a Max-CSP instance  $I$  with hypergraph  $H$  and a branch decomposition of  $\text{inc}(H)$  of MIM-width at most  $k$ , computes  $\text{opt}(I)$  in time polynomial in  $\|I\|$ .*

Let us stress that the results in [38, 34] are given for Max-SAT (and #SAT). However, Theorem 19 can be obtained by adapting the algorithm from [38, 34] to Max-CSPs. We omit the details as Theorem 19 is implied by the results of this section.

The goal of this section is to prove the following:

**Theorem 20.** *Let  $k \geq 1$  be fixed. For every hypergraph  $H$  and branch decomposition of  $\text{inc}(H)$  of MIM-width  $k$ , there exists a point decomposition of  $H$  of polynomial size in  $\|H\|$  and of width at most  $2k$ . Moreover, this point decomposition can be computed in time polynomial in  $\|H\|$ .*

Note that we obtain Theorem 19 as a consequence of Theorem 20 and Theorem 13. In order to prove Theorem 20, we show that the MIM-width of a hypergraph is equivalent to its flat point-width modulo constant factors. This is the main technical result of this section which we state below:

**Theorem 21.** *For every hypergraph  $H$ , we have  $\text{mimw}(H) \leq 4 \cdot \text{fpw}(H)$  and  $\text{fpw}(H) \leq 2 \cdot \text{mimw}(H)$ . Moreover, for a fixed  $k \geq 1$ , a flat point decomposition (of polynomial size) of width at most  $2k$  can be computed in time polynomial in  $\|H\|$  from a branch decomposition of  $H$  of MIM-width  $k$ .*

Note how Theorem 21 directly implies Theorem 20. In order to prove Theorem 21, we present several notions of width and show that they are equivalent modulo constant factors. As an intermediate step, we show a characterisation of the MIM-width of a bipartite graph in terms of its line graph. This characterisation of MIM-width and the one from Theorem 21 may be of independent interest.

## 6.1 A characterisation of the MIM-width of bipartite graphs

A *tree decomposition* of a graph  $G$  is a pair  $(T, (B_t)_{t \in V(T)})$ , where  $T$  is a tree and each bag  $B_t$  is a subset of  $V(G)$  such that

- (i)  $V(G) = \bigcup_{t \in V(T)} B_t$ ,
- (ii) for each edge  $\{u, v\} \in E(G)$ , there exists  $t \in V(T)$  such that  $\{u, v\} \subseteq B_t$ , and
- (iii) for each  $v \in V(G)$  the set  $\{t \in V(T) : v \in B_t\}$  induces a connected subtree of  $T$ .

For any function  $f : 2^{V(G)} \rightarrow \mathbb{Q}_{\geq 0}$ , we define the  $f$ -width of the decomposition  $(T, (B_t)_{t \in V(T)})$  to be the maximum  $f(B_t)$ , taken over all  $t \in V(T)$ , and the  $f$ -width of the graph  $G$  to be the minimum  $f$ -width over all its tree decompositions. For instance, the standard notion of treewidth [32] corresponds to  $s$ -width, where  $s(X) = |X| - 1$ , for every  $X \subseteq V(G)$ .

For a graph  $G$ , we say that a set  $U \subseteq V(G)$  is a *distance-2 independent set* if for every pair of distinct nodes  $u, v \in U$ , there is no path from  $u$  to  $v$  in  $G$  of length at most 2, where the length of a path is the number of edges. We denote by  $\alpha^2(G)$  the maximum size of a distance-2 independent set in  $G$ . For  $G$ , we define the function  $\alpha_G^2 : 2^{V(G)} \rightarrow \mathbb{Q}_{\geq 0}$  as  $\alpha_G^2(X) := \alpha^2(G[X])$ , for every  $X \subseteq V(G)$ . (Recall that  $G[X]$  denotes the subgraph of  $G$  induced by  $X$ , i.e.,  $G[X] = (X, \{\{u, v\} \in E(G) : u, v \in X\})$ .) We also consider the function  $\text{mon-}\alpha_G^2 : 2^{V(G)} \rightarrow \mathbb{Q}_{\geq 0}$  defined by  $\text{mon-}\alpha_G^2(X) := \min\{\alpha_G^2(Y) : X \subseteq Y \subseteq V(G)\}$ , for every  $X \subseteq V(G)$ .

**Observation 22.** *For a graph  $G$ , we have the following:*

- $\alpha_G^2$  is subadditive, i.e.,  $\alpha_G^2(X \cup Y) \leq \alpha_G^2(X) + \alpha_G^2(Y)$ , for all  $X, Y \subseteq V(G)$ .
- $\text{mon-}\alpha_G^2(X) \leq \alpha_G^2(X)$ , for all  $X \subseteq V(G)$ .
- $\text{mon-}\alpha_G^2$  is monotone (unlike  $\alpha_G^2$ ), i.e.,  $\text{mon-}\alpha_G^2(X) \leq \text{mon-}\alpha_G^2(Y)$ , if  $X \subseteq Y \subseteq V(G)$ .

We are particularly interested in the notions of  $\alpha_G^2$ -width and  $\text{mon-}\alpha_G^2$ -width for a graph  $G$ , which we denote by  $\alpha^2\text{-w}(G)$  and  $\text{mon-}\alpha^2\text{-w}(G)$ , respectively. For a graph  $G$ , we define the *line graph* of  $G$ , denoted by  $L(G)$ , to be the graph with vertex set  $E(G)$  such that  $\{e, f\}$  is an edge in  $L(G)$ , where  $e, f \in E(G)$  and  $e \neq f$ , if  $e$  and  $f$  share a common vertex.

**Observation 23.** *Let  $G$  be a graph. Every induced matching in  $G$  is a distance 2-independent set in  $L(G)$  and vice versa. In particular,  $\text{MIM}(G) = \alpha^2(L(G))$ .*

Below we show that for bipartite graphs, the MIM-width and the  $\alpha^2\text{-w}$  (and also  $\text{mon-}\alpha^2\text{-w}$ ) of the line graph are equivalent, modulo constant factors. The proof is an adaptation of the classical equivalence between treewidth and *branchwidth* [33].

**Proposition 24.** *For every graph  $G$ , we have  $\alpha^2\text{-w}(L(G)) \leq 2 \cdot \text{mimw}(G)$ .*

*Proof.* Given a branch decomposition  $(T, \delta)$  of  $G$  of MIM-width  $k$ , we define a tree decomposition of  $L(G)$  of  $\alpha_{L(G)}^2$ -width at most  $2k$ . Recall that for a node  $t \in V(T)$ , we denote by  $T_t$  the subtree of  $T$  rooted at  $t$  and by  $V_t$  the set  $\{\delta^{-1}(\ell) : \ell \text{ is a leaf of } T_t\}$ . The underlying tree of our sought decomposition is  $T$  itself. For  $t \in V(T)$ , we define  $C_t$  to be the set of edges of  $G$  appearing in the bipartite graph  $G[V_t, V(G) \setminus V_t]$ . Now we define  $B_t$  to be  $B_t := C_t$ , if  $t \in V(T)$  is a leaf of  $T$ , and  $B_t := C_t \cup (C_{t_1} \cap C_{t_2})$ , otherwise, where  $t_1$  and  $t_2$  are the two children of  $t$  in  $T$ . We claim that  $(T, (B_t)_{t \in V(T)})$  satisfies the required conditions.

For condition (i) of tree decompositions, for every  $e = \{u, v\} \in E(G) = V(L(G))$ , we have  $e \in B_{\delta(u)}$ . For condition (ii), if  $\{e, f\} \in E(L(G))$  and  $e \cap f = \{u\}$ , then we have  $\{e, f\} \subseteq B_{\delta(u)}$ . In order to prove condition (iii), we show the following properties:

1. Suppose  $e \in E(G) = V(L(G))$  and  $t, t', t'' \in V(T)$  are distinct nodes such that  $t$  is a descendent of  $t'$ ,  $t''$  belongs the (unique) path in  $T$  from  $t$  to  $t'$ , and  $e \in C_t \cap C_{t'}$ . Then  $e \in C_{t''}$ .
2. Suppose  $e \in E(G) = V(L(G))$  and  $t, t', s \in V(T)$  are distinct nodes such that  $t$  and  $t'$  are incomparable in  $T$ ,  $s$  is the least common ancestor of  $t$  and  $t'$  in  $T$ , and  $e \in C_t \cap C_{t'}$ . Then  $e \in C_{s_1} \cap C_{s_2}$ , where  $s_1$  and  $s_2$  are the two children of  $s$  in  $T$ .

For property 1), suppose  $e = \{u, v\}$ , and note that by definition of the  $C_t$ 's, one endpoint of  $e$  belongs to  $V_t$ , say  $u$ , and the other endpoint  $v$  is in  $V(G) \setminus V_{t'}$ . In particular,  $u \in V_{t''}$  and  $v \in V(G) \setminus V_{t''}$ , and hence  $e \in C_{t''}$ . For property 2), let  $e = \{u, v\}$  and note again, by definition of the  $C_t$ 's, that one endpoint of  $e$  belongs to  $V_t$ , say  $u$ , and the other endpoint  $v$  belongs to  $V_{t'}$ . Then if  $s_1$  is the ancestor of  $t$ , we have  $u \in V_{s_1}$  and  $v \in V(G) \setminus V_{s_1}$ , and therefore  $e \in C_{s_1}$ . Similarly for  $s_2$  and  $t'$ .

Now for condition (iii), let  $e \in E(G)$  and  $t, t', t''$  be distinct nodes in  $T$  such that  $t''$  belongs to the (unique) path in  $T$  from  $t$  to  $t'$  and  $e \in B_t \cap B_{t'}$ . We start with the case when  $t$  is a descendent of  $t'$  (the case when  $t'$  is a descendent of  $t$  is analogous). Assume first that  $e \in C_{t'} \subseteq B_{t'}$ . We obtain that  $e \in C_{t''} \subseteq B_{t''}$ , by applying property 1) to  $t', t''$  and either  $t$  (if  $e \in C_t$ ) or a child of  $t$  (if  $e \in B_t \setminus C_t$ ). Suppose now that  $e \in B_{t'} \setminus C_{t'}$ . If  $t''$  is a child of  $t'$ , then  $e \in C_{t''} \subseteq B_{t''}$  and we are done. Otherwise, if  $t'_1$  is the child of  $t'$  that is ancestor of  $t''$ , we obtain  $e \in C_{t''} \subseteq B_{t''}$  by applying property 1) to  $t'_1, t''$  and either  $t$  or a child of  $t$ .

For the case when  $t$  and  $t'$  are incomparable, we let  $s \in V(T)$  be the least common ancestor of  $t$  and  $t'$  in  $T$ . We obtain that  $e \in C_{s_1} \cap C_{s_2} \subseteq B_s$ , where  $s_1$  and  $s_2$  are the two children of  $s$ , by applying property 2) to  $s$ , either  $t$  or one of its child, and either  $t'$  or one of its child (depending on whether  $e \in C_t$  and  $e \in C_{t'}$ , respectively). If  $t'' \neq s$ , we can apply the previous case and obtain that  $e \in B_{t''}$  as required.

It remains to bound the  $\alpha_{L(G)}^2$ -width of  $(T, (B_t)_{t \in V(T)})$ . If  $t \in V(T)$  is a leaf of  $T$ , then  $\alpha_{L(G)}^2(B_t) = 1$ . Otherwise let  $t_1, t_2$  be the two children of  $t$  in  $T$ . By Observation 23, we have  $\alpha_{L(G)}^2(C_t) \leq k$ . By subadditivity, we have that  $\alpha_{L(G)}^2(B_t) \leq \alpha_{L(G)}^2(C_t) + \alpha_{L(G)}^2(C_{t_1} \cap C_{t_2})$ . Observe that  $C_{t_1} \cap C_{t_2} = E(G[V_{t_1}, V_{t_2}])$  (in particular,  $L(G)[C_{t_1} \cap C_{t_2}] = L(G[V_{t_1}, V_{t_2}])$ ). By Observation 23,  $\alpha_{L(G)}^2(C_{t_1} \cap C_{t_2}) = \text{MIM}(G[V_{t_1}, V_{t_2}])$ , and since  $G[V_{t_1}, V_{t_2}]$  is an induced subgraph of  $G[V_{t_1}, V(G) \setminus V_{t_1}]$ , we have  $\text{MIM}(G[V_{t_1}, V_{t_2}]) \leq \text{MIM}(G[V_{t_1}, V(G) \setminus V_{t_1}]) \leq k$ . We obtain that  $\alpha_{L(G)}^2(B_t) \leq 2k$  as required.  $\square$

**Proposition 25.** *For every bipartite graph  $G$ , we have  $\text{mimw}(G) \leq 2 \cdot \text{mon-}\alpha^2\text{-w}(L(G))$ .*

*Proof.* Let  $G$  and  $(T, (B_t)_{t \in V(T)})$  be a tree decomposition of  $L(G)$  of  $\text{mon-}\alpha_{L(G)}^2$ -width  $k$ . We can assume that  $T$  is a binary rooted tree and that there is a bijection  $\delta$  from  $V(G)$  to the leaves of  $T$  such that  $B_{\delta(v)} = \{\{v, w\} \in E(G) : w \in V(G)\}$ , for every  $v \in V(G)$ . To see this, we start by rooting  $(T, (B_t)_{t \in V(T)})$  arbitrarily. For each  $v \in V(G)$ , the set  $\{\{v, w\} \in E(G) : w \in V(G)\}$  is a clique in  $L(G)$ , and hence there exists  $t \in V(T)$  such that  $\{\{v, w\} \in E(G) : w \in V(G)\} \subseteq B_t$ . We add a fresh leaf  $\delta(v)$  to  $T$  as a child of  $t$  and we let  $B_{\delta(v)} := \{\{v, w\} \in E(G) : w \in V(G)\}$ . After this, we iteratively remove all leaves of  $T$  that are not of the form  $\delta(v)$ . Since  $\text{mon-}\alpha_{L(G)}^2(B_{\delta(v)}) = 1$ , for every  $v \in V(G)$ , the width of the resulting decomposition is at most  $k$ . Finally, if a node  $t$  has  $\ell$  children  $t_1, \dots, t_\ell$  with  $\ell > 2$ , we force  $t$  to have only two children  $t_1$  and  $t'$ , where  $t'$  is a fresh node with  $B_{t'} := B_t$  and with children  $t_2, \dots, t_\ell$ . By applying this modification iteratively, we obtain a rooted binary tree as required.

We claim that  $(T, \delta)$  is a branch decomposition of  $G$  of MIM-width at most  $2k$ . Fix  $t \in V(T)$ . We have that  $E(G[V_t, \bar{V}_t]) \subseteq B_t$ , where  $\bar{V}_t := V(G) \setminus V_t$ . Indeed, for  $e = \{u, v\} \in E(G[V_t, \bar{V}_t])$ , we have  $e \in B_{\delta(u)} \cap B_{\delta(v)}$ , and by connectivity,  $e \in B_t$ . Let  $V_1, V_2$  be independent sets partitioning  $V(G)$  (recall that  $G$  is bipartite). Let  $M \subseteq E(G[V_t, \bar{V}_t])$  be a maximum size induced matching in  $G[V_t, \bar{V}_t]$ . Note that  $M$  is the disjoint union of  $M_1$  and  $M_2$ , where  $M_1 = M \cap E(G[V_t \cap V_1, \bar{V}_t \cap V_2])$  and  $M_2 = M \cap E(G[V_t \cap V_2, \bar{V}_t \cap V_1])$ . Finally, observe that

$M_1$  and  $M_2$  are distance 2-independent sets in  $L(G)$  as  $V_1$  and  $V_2$  are independent sets in  $G$ . This implies that  $|M_i| \leq \text{mon-}\alpha^2_{L(G)}(B_t)$ , for  $i \in \{1, 2\}$ . Hence  $|M| \leq 2k$ .  $\square$

By Propositions 24 and 25, for every bipartite graph  $G$ , we have:

$$\frac{1}{2} \cdot \text{mimw}(G) \leq \text{mon-}\alpha^2\text{-w}(L(G)) \leq \alpha^2\text{-w}(L(G)) \leq 2 \cdot \text{mimw}(G).$$

**Remark 26.** As in the case of treewidth, the widths  $\alpha^2\text{-w}$  and  $\text{mon-}\alpha^2\text{-w}$  can be related with other notions such as brambles and games. For instance,  $\alpha^2\text{-w}$  and  $\text{mon-}\alpha^2\text{-w}$  can be lower bounded by the (natural adaptation of the) *bramble number* [36]. Also,  $\text{mon-}\alpha^2\text{-w}$  can be characterised in terms of the monotone version of the *cops and robber game* [36] (this is the reason why we work explicitly with  $\text{mon-}\alpha^2\text{-w}$  in the first place). Now the cops are not restricted to play on a set  $X$  of size  $k$ , but on a set  $X$  with  $\text{mon-}\alpha^2\text{-w}(X) \leq k$ . The minimum  $k$  for which the cops can win the game in a monotone way is precisely the  $\text{mon-}\alpha^2\text{-w}$  (this follows for instance from [1, Theorem 2.2.12 and Remark 2.1.18]). Hence these connections could be used to obtain bounds on the mimw of bipartite graphs.

## 6.2 Proof of Theorem 21

We now show the equivalence of fpw and mimw. Let us start with a definition.

**Definition 27** (Simplified point decomposition). *A simplified point decomposition of a hypergraph  $H$  is a pair  $(T, (B_t)_{t \in V(T)})$  where  $T$  is a rooted tree, each set  $B_t \subseteq P(H)$  is a set of points of  $H$  and*

- (1) *For every edge  $e \in H$ , there exists  $t \in V(T)$  such that  $P(\{e\}) = \{(v, e) : v \in e\} \subseteq B_t$ .*
- (2) *For every subhypergraph  $H'$  of  $H$ , and  $v \in V(H')$ , the set  $\{t \in V(T) : v \in V(H'|_{B_t})\}$  induces a connected subtree of  $T$ .*

As before, the width of a simplified point decomposition  $(T, (B_t)_{t \in V(T)})$  is  $\max_{t \in V(T)} \beta\text{-cn}(H|_{B_t})$ , and the simplified point-width of  $H$ , denoted by  $\text{spw}(H)$ , is the minimum width over all its simplified point decompositions.

**Proposition 28.** *For every hypergraph  $H$ , we have  $\text{fpw}(H) = \text{spw}(H)$ .*

*Proof.* We start by showing  $\text{fpw}(H) \leq \text{spw}(H)$ . Let  $(T, (B_t)_{t \in V(T)})$  be a simplified point decomposition of  $H$  of width  $k$ . We say that two sub-bags  $(t, S)$  and  $(t', S')$  with  $t \neq t'$  are *consistent* if there exists a subhypergraph  $H'$  of  $H$  such that  $S = V(H'|_{B_t})$  and  $S' = V(H'|_{B_{t'}})$ . Consider the triple  $(T, (B_t)_{t \in V(T)}, A)$ , where  $((t, S), (t', S'))$  is an arc in  $A$  if and only if  $t'$  is the parent of  $t$  in  $T$  and,  $(t, S)$  and  $(t', S')$  are consistent. We claim that  $(T, (B_t)_{t \in V(T)}, A)$  is a flat point decomposition of  $H$ , and hence  $\text{fpw}(H) \leq k$ . Let  $H'$  be a subhypergraph of  $H$  and note that if  $t'$  is the parent of  $t$  in  $T$  then there is an arc from  $(t, V(H'|_{B_t}))$  to  $(t', V(H'|_{B_{t'}}))$  in  $A$  as they are consistent. Hence  $A[H']_\emptyset$  (actually we have  $A[H']_\emptyset = A[H']$ ) is a realisation of  $A$ .

Now let  $A'$  be an arbitrary realisation of  $A$ . By definition of  $A$ , we have that the subtree  $T_{A'}$  associated with  $A'$  is actually a subtree of  $T$  that contains the root. By contradiction, suppose the connectivity condition fails for some  $v \in \bigcup_{(t, S) \in V(A')} S$ . Then, there exists a sequence  $(t_0, S_0), \dots, (t_n, S_n)$ , with  $n \geq 2$ , such that (i) each  $(t_i, S_i) \in V(A')$ , (ii)  $t_0, \dots, t_n$

is a path in  $T$ , and (iii)  $v \in S_0 \cap S_n$  but  $v \notin S_i$ , for  $0 < i < n$ . We show by induction that for all  $i \in \{1, \dots, n\}$ , there exists a subhypergraph  $H_i$  of  $H$  such that  $v \in V(H_i|_{B_{t_0}})$ ,  $v \notin V(H_i|_{B_{t_i}})$  and  $S_i \subseteq V(H_i|_{B_{t_i}})$ . In particular,  $v \notin V(H_n|_{B_{t_n}})$  and  $S_n \subseteq V(H_n|_{B_{t_n}})$ . This is a contradiction since  $v \in S_n$ .

For the base case, recall that by construction of  $A$ ,  $(t_0, S_0)$  is consistent with  $(t_1, S_1)$ , and similarly,  $(t_1, S_1)$  with  $(t_2, S_2)$ . Hence, there are subhypergraphs  $H'_0$  and  $H'_1$  of  $H$  such that  $S_0 = V(H'_0|_{B_{t_0}})$ ,  $S_1 = V(H'_0|_{B_{t_1}}) = V(H'_1|_{B_{t_1}})$  and  $S_2 = V(H'_1|_{B_{t_2}})$ . We define  $H_1 = H'_0 \cup H'_1$ . Then we have that  $S_0 \subseteq V(H_1|_{B_{t_0}})$  and  $S_1 = V(H_1|_{B_{t_1}})$ . In particular,  $v \in S_0 \subseteq V(H_1|_{B_{t_0}})$ ,  $v \notin S_1 = V(H_1|_{B_{t_1}})$  and  $S_1 \subseteq V(H_1|_{B_{t_1}})$ , as required. For the inductive case, suppose we have  $H_i$  with the desired properties, for  $i \in \{1, \dots, n-1\}$ . As  $(t_i, S_i)$  and  $(t_{i+1}, S_{i+1})$  are consistent, there is a subhypergraph  $H'_i$  of  $H$  such that  $S_i = V(H'_i|_{B_{t_i}})$  and  $S_{i+1} = V(H'_i|_{B_{t_{i+1}}})$ . We take  $H_{i+1} = H_i \cup H'_i$ . Note that  $S_{i+1} \subseteq V(H_{i+1}|_{B_{t_{i+1}}})$  and  $v \in V(H_{i+1}|_{B_{t_0}})$  (using the inductive hypothesis  $v \in V(H_i|_{B_{t_0}})$ ). Observe that  $V(H_{i+1}|_{B_{t_i}}) = V(H_i|_{B_{t_i}}) \cup S_i$ . Since  $v \notin S_i$  and  $v \notin V(H_i|_{B_{t_i}})$  (by inductive hypothesis), we derive that  $v \notin V(H_{i+1}|_{B_{t_i}})$ . Since  $v \in V(H_{i+1}|_{B_{t_0}})$ , it follows that  $v \notin V(H_{i+1}|_{B_{t_{i+1}}})$ ; otherwise the connectivity condition (2) for simplified point decompositions would be violated for  $H_{i+1}$ . Hence  $H_{i+1}$  satisfies all the required conditions.

For  $\text{fpw}(H) \geq \text{spw}(H)$ , let  $(T, (B_t)_{t \in V(T)}, A)$  be a flat point decomposition of  $H$  of width  $k$ . We claim that  $(T, (B_t)_{t \in V(T)})$  is a simplified point decomposition of  $H$ , and the result follows. Let  $H'$  be a subhypergraph of  $H$ . By definition of point decompositions,  $A[H']_\emptyset$  is a realisation of  $A$  and for every  $v \in V(H')$ , the set  $\{t \in V(T_{A[H']_\emptyset}) : v \in V(H'|_{B_t})\}$  induces a connected subtree of  $T_{A[H']_\emptyset}$ . For every  $t \in V(T) \setminus V(T_{A[H']_\emptyset})$ , we have  $V(H'|_{B_t}) = \emptyset$  and then  $\{t \in V(T_{A[H']_\emptyset}) : v \in V(H'|_{B_t})\} = \{t \in V(T) : v \in V(H'|_{B_t})\}$ . Since  $T_{A[H']_\emptyset}$  must be a subtree of  $T$ , the latter set induces a connected subtree of  $T$ . Hence condition (2) of Definition 27 (simplified point decompositions) holds.  $\square$

Observe how a simplified point decomposition of  $H$  encodes tree decompositions for the subhypergraphs of  $H$  without the need of a  $\mathcal{T}$ -structure, unlike the case of flat point decompositions. Whether arbitrary point decompositions can also be captured by a notion of decomposition that does not use  $\mathcal{T}$ -structures explicitly is an interesting question which we leave for future work.

For a hypergraph  $H$ , we define the *point graph* of  $H$ , denoted by  $\text{pg}(H)$ , as  $\text{pg}(H) := (P(H), \{\{(v, e), (v', e')\} : v = v' \text{ or } e = e'\})$ . Note that the point graph  $\text{pg}(H)$  of  $H$  is isomorphic to  $L(\text{inc}(H))$ . There is a known duality between  $\beta$ -cn and MIM (see e.g. [6, Theorem 2.18]):

**Observation 29.** *For every hypergraph  $H$ , we have  $\beta\text{-cn}(H) = \text{MIM}(\text{inc}(H))$ . By Observation 23, we have  $\beta\text{-cn}(H) = \alpha^2(\text{pg}(H))$ .*

**Proposition 30.** *For every hypergraph  $H$ , we have  $\text{spw}(H) \leq \alpha^2\text{-w}(\text{pg}(H))$  and  $\alpha^2\text{-w}(\text{pg}(H)) \leq 2 \cdot \text{spw}(H)$ .*

*Proof.* For  $\text{spw}(H) \leq \alpha^2\text{-w}(\text{pg}(H))$ , let  $(T, (B_t)_{t \in V(T)})$  be a tree decomposition of  $\text{pg}(H)$  of  $\alpha^2$ -width  $k$ . We claim that  $(T, (B_t)_{t \in V(T)})$  is a simplified point decomposition of  $H$  of width  $k$ . By Observation 29, we have  $\beta\text{-cn}(H|_{B_t}) = \alpha^2(\text{pg}(H|_{B_t})) = \alpha^2(\text{pg}(H)[B_t]) = \alpha^2_{\text{pg}(H)}(B_t)$ , for every  $t \in V(T)$ . Hence, the width of  $(T, (B_t)_{t \in V(T)})$  is  $k$ . For condition (1) of Definition 27, let  $e \in H$  and note that the set  $\{(v, e) \in P(H) : v \in e\}$  forms a clique in  $\text{pg}(H)$ . Hence, there exists  $t \in V(T)$  such that  $\{(v, e) \in P(H) : v \in e\} \subseteq B_t$ . Towards a contradiction, suppose

that condition (2) of Definition 27 is violated, i.e., there is a subhypergraph  $H'$  of  $H$ , a vertex  $v \in V(H')$  and distinct nodes  $t_1, t_2, t_3 \in V(T)$  such that  $t_3$  is in the unique path from  $t_1$  to  $t_2$  in  $T$ , and  $v \in V(H'|_{B_{t_1}}) \cap V(H'|_{B_{t_2}})$  but  $v \notin V(H'|_{B_{t_3}})$ . In particular, there exist edges  $e_1, e_2 \in H'$  such that  $(v, e_1) \in B_{t_1}$ ,  $(v, e_2) \in B_{t_2}$  and  $\{(v, e_1), (v, e_2)\} \cap B_{t_3} = \emptyset$ . Since  $\{(v, e_1), (v, e_2)\}$  is an edge in  $pg(H)$ , there is a node  $t \in V(T)$  such that  $\{(v, e_1), (v, e_2)\} \subseteq B_t$ . Using the connectivity of the tree decomposition  $(T, (B_t)_{t \in V(T)})$ , we obtain that  $\{(v, e_1), (v, e_2)\} \cap B_{t_3} \neq \emptyset$ ; a contradiction.

For  $\alpha^2\text{-w}(pg(H)) \leq 2 \cdot \text{spw}(H)$ , let  $(T, (B_t)_{t \in V(T)})$  be a simplified point decomposition of  $H$  of width  $k$ . We define  $T'$  to be the tree obtained from  $T$  by subdividing every edge in  $E(T)$ , i.e., replacing every edge  $e = \{t_1, t_2\} \in E(T)$  by two edges  $\{t_1, t_e\}$  and  $\{t_e, t_2\}$ , where  $t_e$  is a fresh node. For  $t \in V(T')$ , we define  $B'_t := B_t$ , if  $t \in V(T)$ , or  $B'_t := B_{t_1} \cup B_{t_2}$ , if  $t = t_e$  with  $e = \{t_1, t_2\}$ .

We claim that  $(T', (B'_t)_{t \in V(T')})$  is a tree decomposition of  $pg(H)$ . First note that, for every point  $(v, e)$  in  $H$ , by condition (1) of simplified point decompositions, there is  $t \in V(T) \subseteq V(T')$ , such that  $(v, e) \in B_t = B'_t$ , and hence condition (i) of tree decompositions holds. For condition (ii), suppose  $(v, e)$  and  $(v', e)$  are points with  $v \neq v'$ . Again by condition (1), we obtain that there is  $t \in V(T) \subseteq V(T')$ , such that  $\{(v, e), (v', e)\} \in B_t = B'_t$ . Now suppose that  $(v, e)$  and  $(v, e')$  are points with  $e \neq e'$  and pick  $t, t' \in V(T)$  such that  $(v, e) \in B_t$  and  $(v, e') \in B_{t'}$ . By applying condition (2) of simplified point decompositions to the subhypergraph  $H' = \{e, e'\}$ , we have that  $\{(v, e), (v, e')\} \cap B_s \neq \emptyset$ , for every  $s \in V(T)$  in the unique path from  $t$  to  $t'$  in  $T$ . In particular, there is an edge  $\hat{e} = \{s_1, s_2\}$  in this path such that  $(v, e) \in B_{s_1}$  and  $(v, e') \in B_{s_2}$ . It follows that  $\{(v, e), (v, e')\} \subseteq B'_{t_{\hat{e}}}$ , for  $t_{\hat{e}} \in V(T')$ , and hence condition (ii) holds. For a point  $(v, e)$  of  $H$ , condition (iii) follows from applying condition (2) to the subhypergraph  $H' = \{e\}$ . Finally, note that, by Observation 29 and subadditivity of  $\alpha^2_{pg(H)}$ , the  $\alpha^2_{pg(H)}$ -width of  $(T', (B'_t)_{t \in V(T')})$  is at most  $2k$ , as required.  $\square$

Theorem 21 follows from Propositions 30, 28, 24, and 25. Let us stress that given a branch decomposition  $(T, \delta)$  of  $inc(H)$  of MIM-width  $k \geq 1$ , we can efficiently compute a flat point decomposition (of polynomial size) of width at most  $2k$ . By applying the construction in the proof of Proposition 24 (and due to Proposition 30), from  $(T, \delta)$  we can efficiently compute a simplified point decomposition for  $H$  of width at most  $2k$ . Finally, the construction in the proof of Proposition 28 of a flat point decomposition from the simplified point decomposition of width  $2k$ , in particular, of the  $\mathcal{T}$ -structure  $A$ , can be done in polynomial time. The main step is given two nodes  $t, t' \in V(T)$ , where  $t'$  is the parent of  $t$ , and two sub-bags of the form  $(t, S_1)$  and  $(t', S_2)$ , to check whether they are consistent. This is equivalent to checking the existence of two subhypergraphs  $H_1$  and  $H_2$  with  $|H_1| \leq 2k$ ,  $|H_2| \leq 2k$ , such that (i)  $S_1 = V(H_1|_{B_t})$ ,  $S_2 = V(H_2|_{B_{t'}})$ , and (ii)  $V(H_1|_{B_{t'}}) \subseteq S_2$  and  $V(H_2|_{B_t}) \subseteq S_1$ . This can be checked in polynomial time.

## 7 Conclusions

We have introduced a new width that unifies  $\beta$ -acyclicity and bounded MIM-width. We have also identified a novel island of tractability for structurally restricted Max-CSPs. The main open problem is to obtain more general hypergraph properties that lead to tractability, and ultimately find the precise boundary of tractability. There are many natural hypergraph properties that generalise bounded point-width whose tractability status is unclear (from less to more general):



bounded  $\beta$ -hypertreewidth ( $\beta$ -hw) [21], bounded  $\beta$ -fractional hypertreewidth ( $\beta$ -fhw), and bounded  $\beta$ -submodular width ( $\beta$ -subw). In particular, we have  $\beta$ -subw  $\leq$   $\beta$ -fhw  $\leq$   $\beta$ -hw  $\leq$  pw. For precise definitions, see Appendix A.

In addition to  $\beta$ -acyclicity and MIM-width, our notion of point-width also subsumes a width measure called *coverwidth*, introduced in [6, Section 5.3.2]. In Appendix D, we show that every class of hypergraphs of bounded coverwidth also has bounded flat point-width, and hence, bounded MIM-width. We also show that the converse does not hold, i.e., bounded MIM-width strictly generalises bounded coverwidth.

We have focused on polynomial-time solvability for Max-CSPs. Regarding *fixed-parameter tractability* (FPT), it is easy to show (cf. Appendix B) that Marx’s classification of CSPs [29] implies an FPT classification of  $\{0,1\}$ -valued Max-CSPs and the FPT frontier is given by the classes with bounded  $\beta$ -submodular width. This classification implies that for a class of unbounded  $\beta$ -submodular width the  $\{0,1\}$ -valued, and hence the finite-valued, problem Max-CSP( $\mathcal{H}, -$ ) is not fixed-parameter (and thus not polynomial-time) tractable. Note that a collapse between bounded point-width and bounded  $\beta$ -submodular width would give us a complete classification of Max-CSPs in terms of polynomial time-solvability (and FPT). Hence, a natural research direction is to study the relationship between all these measures (pw,  $\beta$ -hw,  $\beta$ -fhw and  $\beta$ -subw). As a related result, which could be interesting in its own right, we show (cf. Appendix C) that bounded  $\beta$ -fractional hypertreewidth collapses to bounded  $\beta$ -hypertreewidth.

We finish with a few open problems that were mentioned throughout the paper. Firstly, we have shown (Theorem 18) that every  $\beta$ -acyclic hypergraph has a point decomposition of polynomial size and width 1. We do not know whether the converse is true. Secondly, as discussed before Example 5 in Section 3, we do not know whether the problem of checking that a given triple is a point decomposition admits an efficient algorithm. Lastly, our algorithm is only polynomial-time if the input point decomposition has polynomial size (Theorem 13). Is it true that for any fixed  $k > 0$ , every hypergraph of point-width  $k$  has a polynomial-sized point decomposition of width  $f(k)$  for some function  $f$ ?

TODO:  $\beta$ -acyclic and bounded MIM-width do not (or do they?) have to come up with decompositions.

## Acknowledgements

We would like to thank the anonymous referees of both the conference [9] and this full version of the paper.

## References

- [1] I. Adler, Width functions for hypertree decompositions, Ph.D. thesis, Albert Ludwig University of Freiburg (2006).
- [2] C. Beeri, R. Fagin, D. Maier, A. Mendelzon, J. Ullman, M. Yannakakis, Properties of acyclic database schemes, in: Proceedings of the 13th Annual ACM Symposium on Theory of Computing (STOC’81), 1981, pp. 355–362. doi:10.1145/800076.802489.
- [3] C. Beeri, R. Fagin, D. Maier, M. Yannakakis, On the desirability of acyclic database schemes, Journal of the ACM 30 (3) (1983) 479–513. doi:10.1145/2402.322389.

- [4] J. Brault-Baron, F. Capelli, S. Mengel, Understanding model counting for beta-acyclic CNF-formulas, in: Proceedings of the 32nd International Symposium on Theoretical Aspects of Computer Science (STACS'15), 2015, pp. 143–156. [doi:10.4230/LIPIcs.STACS.2015.143](https://doi.org/10.4230/LIPIcs.STACS.2015.143).
- [5] A. Bulatov, A dichotomy theorem for nonuniform CSPs, in: Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17), IEEE, 2017, pp. 319–330. [doi:10.1109/FOCS.2017.37](https://doi.org/10.1109/FOCS.2017.37).
- [6] F. Capelli, Structural restrictions of CNF-formulas: applications to model counting and knowledge compilation, Ph.D. thesis, Université Paris Diderot (2016).
- [7] F. Capelli, Understanding the complexity of #SAT using knowledge compilation, in: Proceedings of the 32nd Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'17), 2017, pp. 1–10. [doi:10.1109/LICS.2017.8005121](https://doi.org/10.1109/LICS.2017.8005121).
- [8] C. Carbonnel, M. Romero, S. Živný, The complexity of general-valued CSPs seen from the other side, in: Proceedings of the 59th Annual IEEE Symposium on Foundations of Computer Science (FOCS'18), IEEE, 2018, pp. 319–330. [doi:10.1109/FOCS.2018.00031](https://doi.org/10.1109/FOCS.2018.00031).
- [9] C. Carbonnel, M. Romero, S. Živný, Point-width and Max-CSPs, in: Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS'19), 2019.
- [10] A. K. Chandra, P. M. Merlin, Optimal implementation of conjunctive queries in relational data bases, in: Proceedings of the 9th Annual ACM Symposium on Theory of Computing (STOC'77), ACM, 1977, pp. 77–90. [doi:10.1145/800105.803397](https://doi.org/10.1145/800105.803397).
- [11] H. Chen, M. Grohe, Constraint satisfaction with succinctly specified relations, *Journal of Computer and System Sciences* 76 (8) (2010) 847–860. [doi:10.1016/j.jcss.2010.04.003](https://doi.org/10.1016/j.jcss.2010.04.003).
- [12] V. Dalmau, P. G. Kolaitis, M. Y. Vardi, Constraint Satisfaction, Bounded Treewidth, and Finite-Variable Logics, in: Proceedings of the 8th International Conference on Principles and Practice of Constraint Programming (CP'02), Vol. 2470 of Lecture Notes in Computer Science, Springer, 2002, pp. 310–326. [doi:10.1007/3-540-46135-3\\_21](https://doi.org/10.1007/3-540-46135-3_21).
- [13] R. Diestel, *Graph Theory*, Fourth Edition, Springer, 2010.
- [14] R. Fagin, Degrees of Acyclicity for Hypergraphs and Relational Database Schemes, *Journal of the ACM* 30 (1983) 514–550. [doi:10.1145/2402.322390](https://doi.org/10.1145/2402.322390).
- [15] T. Feder, M. Y. Vardi, The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory, *SIAM Journal on Computing* 28 (1) (1998) 57–104. [doi:10.1137/S0097539794266766](https://doi.org/10.1137/S0097539794266766).
- [16] W. Fischl, G. Gottlob, R. Pichler, General and fractional hypertree decompositions: Hard and easy cases, in: Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS'18), 2018, pp. 17–32. [doi:10.1145/3196959.3196962](https://doi.org/10.1145/3196959.3196962).
- [17] A. Frank, Some polynomial algorithms for certain graphs and hypergraphs, in: Proceedings of the 5th British Combinatorial Conference, 1975, *Utilitas Mathematica*, 1975.

- [18] E. C. Freuder, Complexity of K-Tree Structured Constraint Satisfaction Problems, in: Proceedings of the 8th National Conference on Artificial Intelligence (AAAI'90), 1990, pp. 4–9.
- [19] G. Gottlob, G. Greco, F. Scarcello, Tractable Optimization Problems through Hypergraph-Based Structural Restrictions, in: Proceedings of the 36th International Colloquium on Automata, Languages and Programming (ICALP'09), Part II, Vol. 5556 of Lecture Notes in Computer Science, Springer, 2009, pp. 16–30. doi:[10.1007/978-3-642-02930-1\\_2](https://doi.org/10.1007/978-3-642-02930-1_2).
- [20] G. Gottlob, N. Leone, F. Scarcello, Hypertree decomposition and tractable queries, Journal of Computer and System Sciences 64 (3) (2002) 579–627. doi:[10.1006/jcss.2001.1809](https://doi.org/10.1006/jcss.2001.1809).
- [21] G. Gottlob, R. Pichler, Hypergraphs in model checking: Acyclicity and hypertree-width versus clique-width, SIAM J. Comput. 33 (2) (2004) 351–378. doi:[10.1137/S0097539701396807](https://doi.org/10.1137/S0097539701396807).
- [22] M. Grohe, The complexity of homomorphism and constraint satisfaction problems seen from the other side, Journal of the ACM 54 (1) (2007) 1–24. doi:[10.1145/1206035.1206036](https://doi.org/10.1145/1206035.1206036).
- [23] M. Grohe, D. Marx, Constraint solving via fractional edge covers, ACM Transactions on Algorithms 11 (1) (2014) 4:1–4:20. doi:[10.1145/2636918](https://doi.org/10.1145/2636918).
- [24] M. Grohe, T. Schwentick, L. Segoufin, When is the evaluation of conjunctive queries tractable?, in: Proceedings of the 33th Annual ACM Symposium on Theory of Computing (STOC'01), 2001, pp. 657–666. doi:[10.1145/380752.380867](https://doi.org/10.1145/380752.380867).
- [25] P. Hell, J. Nešetřil, On the Complexity of  $H$ -coloring, Journal of Combinatorial Theory, Series B 48 (1) (1990) 92–110. doi:[10.1016/0095-8956\(90\)90132-J](https://doi.org/10.1016/0095-8956(90)90132-J).
- [26] P. Hell, J. Nešetřil, Graphs and Homomorphisms, Oxford University Press, 2004.
- [27] P. G. Jeavons, On the Algebraic Structure of Combinatorial Problems, Theoretical Computer Science 200 (1-2) (1998) 185–204. doi:[10.1016/S0304-3975\(97\)00230-2](https://doi.org/10.1016/S0304-3975(97)00230-2).
- [28] P. G. Kolaitis, M. Y. Vardi, Conjunctive-query containment and constraint satisfaction, in: Proceedings of the 17th SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'98), 1998, pp. 205–213. doi:[10.1145/275487.275511](https://doi.org/10.1145/275487.275511).
- [29] D. Marx, Tractable hypergraph properties for constraint satisfaction and conjunctive queries, Journal of the ACM 60 (6), article No. 42. doi:[10.1145/2535926](https://doi.org/10.1145/2535926).
- [30] U. Montanari, Networks of Constraints: Fundamental properties and applications to picture processing, Information Sciences 7 (1974) 95–132. doi:[10.1016/0020-0255\(74\)90008-5](https://doi.org/10.1016/0020-0255(74)90008-5).
- [31] P. Raghavendra, Optimal algorithms and inapproximability results for every CSP?, in: Proceedings of the 40th Annual ACM Symposium on Theory of Computing (STOC'08), 2008, pp. 245–254. doi:[10.1145/1374376.1374414](https://doi.org/10.1145/1374376.1374414).
- [32] N. Robertson, P. D. Seymour, Graph minors. III. Planar tree-width, Journal of Combinatorial Theory, Series B 36 (1) (1984) 49–64. doi:[10.1016/0095-8956\(84\)90013-3](https://doi.org/10.1016/0095-8956(84)90013-3).

- [33] N. Robertson, P. D. Seymour, Graph minors. X. Obstructions to tree-decomposition, *Journal of Combinatorial Theory, Series B* 52 (2) (1991) 153–190. [doi:10.1016/0095-8956\(91\)90061-N](https://doi.org/10.1016/0095-8956(91)90061-N).
- [34] S. H. Sæther, J. A. Telle, M. Vatshelle, Solving #SAT and MAXSAT by dynamic programming, *J. Artif. Intell. Res.* 54 (2015) 59–82. [doi:10.1613/jair.4831](https://doi.org/10.1613/jair.4831).
- [35] T. J. Schaefer, The Complexity of Satisfiability Problems, in: *Proceedings of the 10th Annual ACM Symposium on Theory of Computing (STOC'78)*, ACM, 1978, pp. 216–226. [doi:10.1145/800133.804350](https://doi.org/10.1145/800133.804350).
- [36] P. D. Seymour, R. Thomas, Graph searching and a min-max theorem for tree-width, *Journal of Combinatorial Theory Series B* 58 (1) (1993) 22–33. [doi:10.1006/jctb.1993.1027](https://doi.org/10.1006/jctb.1993.1027).
- [37] J. Thapper, S. Živný, The complexity of finite-valued CSPs, *Journal of the ACM* 63 (4), article No. 37. [doi:10.1145/2974019](https://doi.org/10.1145/2974019).
- [38] M. Vatshelle, New width parameters of graphs, Ph.D. thesis, University of Bergen (2012).
- [39] M. Yannakakis, Algorithms for acyclic database schemes, in: *Proceedings of the 7th International Conference on Very Large Data Bases (VLDB'81)*, IEEE Computer Society, 1981, pp. 82–94.
- [40] D. Zhuk, A proof of CSP dichotomy conjecture, in: *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS'17)*, IEEE, 2017, pp. 331–342. [doi:10.1109/FOCS.2017.38](https://doi.org/10.1109/FOCS.2017.38).

## A Width measures

Let  $H$  be a hypergraph and  $X \subseteq V(H)$ . The hypergraph *induced* by  $X$ , denote by  $H[X]$ , is defined as

$$H[X] := \{e \cap X : e \in H, e \cap X \neq \emptyset\}.$$

Note that, in general,  $H[X]$  is not a subhypergraph of  $H$  as defined in Section 2.

A *fractional edge cover* of a hypergraph  $H$  is a function  $\gamma : H \rightarrow \mathbb{Q}_{\geq 0}$  such that for all  $v \in V(H)$ ,  $\sum_{e \in H: v \in e} \gamma(e) \geq 1$ , and the fractional edge cover number of  $H$ , denoted by  $\text{fcn}(H)$ , is the minimum of  $\sum_{e \in H} \gamma(e)$  over all fractional edge covers  $\gamma$  of  $H$ .

A *tree decomposition* of a hypergraph  $H$  is a pair  $(T, (B_t)_{t \in V(T)})$ , where  $T$  is a tree and each bag  $B_t$  is a subset of  $V(H)$  such that (i) for each  $e \in H$  there exists  $t \in V(T)$  such that  $e \subseteq B_t$  and (ii) for each  $v \in V(H)$  the set  $\{t \in V(T) : v \in B_t\}$  induces a connected subtree of  $T$ .

Let  $H$  be a hypergraph. For any function  $f : 2^{V(H)} \rightarrow \mathbb{Q}_{\geq 0}$ , we define the *f-width* of a tree decomposition  $(T, (B_t)_{t \in V(T)})$  of  $H$  as the maximum of  $f(B_t)$  taken over all  $t \in V(T)$ , and the *f-width of  $H$*  as the minimum *f-width* of a tree decomposition of  $H$ . Given a hypergraph  $H$ ,

- The *treewidth* [32] of  $H$  is its *s-width*, where  $s(X) = |X| - 1$ ;
- The (generalised) *hypertreewidth* [20] of  $H$  is its *c-width*, where  $c(X) = \text{cn}(H[X])$ ;
- The *fractional hypertreewidth* [23] of  $H$  is its *fc-width*, where  $fc(X) = \text{fcn}(H[X])$ .

The treewidth, hypertreewidth and fractional hypertreewidth of a hypergraph  $H$  will be denoted by  $\text{tw}(H)$ ,  $\text{hw}(H)$  and  $\text{fhw}(H)$ , respectively. Let us notice that a hypergraph  $H$  is  $\alpha$ -acyclic if and only if  $\text{hw}(H) = 1$ .

Let  $H$  be a hypergraph. If  $\mathcal{F}$  is a set of functions from  $2^{V(H)}$  to  $\mathbb{Q}_{\geq 0}$ , we call  $\mathcal{F}$ -width of  $H$  the quantity  $\sup\{f\text{-width}(H) : f \in \mathcal{F}\}$ . A function  $f : 2^{V(H)} \rightarrow \mathbb{Q}_{\geq 0}$  is *edge-dominated* if  $f(e) \leq 1$  for all  $e \in H$ , and *submodular* if  $f(A \cap B) + f(A \cup B) \leq f(A) + f(B)$  for all  $A, B \subseteq V(H)$ . The *submodular width* [29] of  $H$ , denoted by  $\text{subw}(H)$ , is its  $\mathcal{F}_s$ -width, where  $\mathcal{F}_s$  is the set of all edge-dominated submodular functions from  $2^{V(H)}$  to  $\mathbb{Q}_{\geq 0}$  satisfying  $f(\emptyset) = 0$ .

Given a hypergraph  $H$ , the  $\beta$ -hypertreewidth [21] (resp.  $\beta$ -fractional hypertreewidth,  $\beta$ -submodular width) of  $H$  is the maximum hypertreewidth (resp. fractional hypertreewidth, submodular width) taken over all subhypergraphs of  $H$ . We denote these quantities by  $\beta\text{-hw}(H)$ ,  $\beta\text{-fhw}(H)$  and  $\beta\text{-subw}(H)$ , respectively. Observe that a hypergraph  $H$  is  $\beta$ -acyclic if and only if  $\beta\text{-hw}(H) = 1$ .

## B FPT Classification for $\{0,1\}$ -valued Max-CSPs

We denote by  $\{0,1\}$ -Max-CSP the restriction of Max-CSP to  $\{0,1\}$ -valued functions. In other words, an instance of  $\{0,1\}$ -Max-CSP is syntactically identical to a CSP instance but the goal is to compute the maximum number of constraints that can be simultaneously satisfied.

We shall consider a parameterised version of  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ) with parameter  $|H|$  (we slightly abuse notation and denote this parameterised problem simply  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ )). In particular,  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ) is in the class FPT of fixed-parameter tractable problems if an instance  $I$  of  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ) can be solved in time  $f(|H|) \cdot |I|^c$ , where  $f$  is any computable function and  $c > 0$  is a constant.

**Theorem 31.** *Let  $\mathcal{H}$  be a recursively enumerable class of hypergraphs. Then, assuming the Exponential Time Hypothesis (ETH),  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ) is in FPT if and only if  $\mathcal{H}$  has bounded  $\beta$ -submodular width.*

*Proof.* For the tractability part, suppose  $\mathcal{H}$  has bounded  $\beta$ -submodular width and let  $I$  be an instance of  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ). Let  $\pi = I_1, \dots, I_r$  be an enumeration of all the sub-instances of  $I$  (that is, instances obtained from  $I$  by removing some constraints) ordered in non-increasing order according to the number of constraints (and hence according to the number of edges in the underlying hypergraph). To compute the optimal value of  $I$ , it suffices to find the first sub-instance according to  $\pi$  that have a solution. Since each sub-instance has bounded submodular width, the whole procedure can be done in FPT [29].

For the hardness, suppose that  $\mathcal{H}$  has unbounded  $\beta$ -submodular width. Then for each  $H \in \mathcal{H}$  we can take a subhypergraph  $H'$  such that the class  $\mathcal{H}' := \{H' \mid H \in \mathcal{H}\}$  has unbounded submodular width. By Marx's result [29], assuming ETH, we have that CSP( $\mathcal{H}', -$ ) is not in FPT. It suffices to show that CSP( $\mathcal{H}', -$ ) fpt-reduces to  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ). Let  $I$  be an instance of CSP( $\mathcal{H}', -$ ) with underlying hypergraph  $H' \in \mathcal{H}'$ . We start by enumerating  $\mathcal{H}$  until we find a hypergraph  $H$  that contains as a subhypergraph  $H'$ . By definition of  $\mathcal{H}'$ , such an  $H$  must exist. Let  $J$  be the instance of  $\{0,1\}$ -Max-CSP( $\mathcal{H}, -$ ) obtained from  $I$  by additionally adding one empty constraint for each edge  $e \in H \setminus H'$ . We have that  $I$  has a solution if and only if the optimal value of  $J$  is the number of constraints in  $I$ . Note that the reduction can be done in FPT time.  $\square$

## C Collapse of $\beta$ -hypertreewidth and $\beta$ -fractional hypertreewidth

It follows from the definitions that  $\beta\text{-fhw}(H) \leq \beta\text{-hw}(H)$ , for every hypergraph  $H$ . In this section we show that  $\beta\text{-hw}(H) \leq f(\beta\text{-fhw}(H))$ , for a fixed function  $f$  (Proposition 34). The key ingredient of the proof is the following lemma, which we borrow from [16]. The *VC dimension* of a hypergraph  $H$ , denoted by  $\text{VC}(H)$ , is the size of the largest set  $X \subseteq V(H)$  such that  $H[X] = 2^X$ . Note that the precise statement of this result as given in [16] (in the proof of Theorem 6.1) differs by a factor  $\text{fcn}(H)$ , but we believe that this is due to a typographical error on their side.

**Lemma 32** ([16]). *For any hypergraph  $H$ , it holds that*

$$\text{cn}(H) \leq 2^{\text{VC}(H)+2} \cdot \text{fcn}(H) \cdot \log(11 \cdot \text{fcn}(H))$$

It follows that if a hypergraph  $H$  has a fractional edge cover of small weight then it has a small edge cover unless its VC dimension is large. We will combine this fact with a straightforward upper bound on the VC dimension in terms of  $\beta\text{-fhw}(H)$ .

**Lemma 33.** *For any hypergraph  $H$ , it holds that*

$$\text{VC}(H) \leq 2 \cdot \beta\text{-fhw}(H)$$

*Proof.* Let  $X \subseteq V(H)$  be a subset of vertices of size  $\text{VC}(H)$  such that  $H[X] = 2^X$ . Let  $K_X$  be the complete graph with vertex set  $X$ . Since  $K_X$  is a subhypergraph of  $H[X]$  and  $H[X]$  is an induced hypergraph of  $H$ , it holds that  $\beta\text{-fhw}(H) \geq \beta\text{-fhw}(H[X]) \geq \text{fhw}(K_X)$ . Now, let  $(T, (B_t)_{t \in V(T)})$  be a tree decomposition of  $K_X$  of  $f$ -width  $\text{fhw}(K_X)$ . For each  $t \in V(T)$ , let  $\gamma_t$  be a fractional edge cover of  $K_X[B_t]$  such that  $\sum_{e \in K_X[B_t]} \gamma_t(e) = \text{fcn}(K_X[B_t])$ . Since  $K_X$  is a clique on  $X$ , there exists  $t^* \in V(T)$  such that  $B_{t^*} = X$ , and hence  $K_X[B_{t^*}] = K_X$ . It follows that

$$\text{fhw}(K_X) \geq \sum_{e \in K_X} \gamma_{t^*}(e) \geq \frac{1}{2} \sum_{v \in X} \sum_{e \in K_X: v \in e} \gamma_{t^*}(e) \geq \frac{1}{2}|X|.$$

Hence,

$$\beta\text{-fhw}(H) \geq \beta\text{-fhw}(H[X]) \geq \text{fhw}(K_X) \geq \frac{1}{2}|X| = \frac{1}{2}\text{VC}(H).$$

□

**Proposition 34.** *For any hypergraph  $H$ , it holds that*

$$\beta\text{-hw}(H) \leq 4^{\beta\text{-fhw}(H)+1} \cdot \beta\text{-fhw}(H) \cdot \log(11 \cdot \beta\text{-fhw}(H)).$$

*Proof.* Let  $H'$  be a subhypergraph of  $H$ , and  $(T, (B_t)_{t \in V(T)})$  be a tree decomposition of  $H'$  of  $f$ -width at most  $\beta\text{-fhw}(H)$ . By Lemma 32 and Lemma 33, for each bag  $B_t$  we have

$$\begin{aligned} \text{cn}(H'[B_t]) &\leq 2^{\text{VC}(H'[B_t])+2} \cdot \text{fcn}(H'[B_t]) \cdot \log(11 \cdot \text{fcn}(H'[B_t])) \\ &\leq 2^{\text{VC}(H)+2} \cdot \beta\text{-fhw}(H) \cdot \log(11 \cdot \beta\text{-fhw}(H)) \\ &\leq 4^{\beta\text{-fhw}(H)+1} \cdot \beta\text{-fhw}(H) \cdot \log(11 \cdot \beta\text{-fhw}(H)) \end{aligned}$$

and hence the hypertreewidth of  $H'$  is at most  $4^{\beta\text{-fhw}(H)+1} \cdot \beta\text{-fhw}(H) \cdot \log(11 \cdot \beta\text{-fhw}(H))$ . This is true for all choices of subhypergraph  $H'$ , so the claim follows. □

**Corollary 35.** *A class of hypergraphs has bounded  $\beta$ -hypertreewidth if and only if it has bounded  $\beta$ -fractional hypertreewidth.*

## D Coverwidth and MIM-width

In this section, we prove that bounded MIM-width strictly generalises bounded coverwidth. We start with some definitions. Let  $H$  be a hypergraph and  $<$  be an ordering of  $V(H)$ . For  $x \in V(H)$ , we define  $H^x$  to be the set of edges of  $H$  that can be reached from  $x$  using only vertices  $\leq x$ . More formally, a walk from  $x \in V(H)$  to  $e \in H$  is a sequence  $(x_1, e_1, x_2, e_2, \dots, x_n, e_n)$  with  $n \geq 1$  such that  $x = x_1$ ,  $e = e_n$ ,  $x_n \in e_n$  and  $\{x_i, x_{i+1}\} \subseteq e_i$ , for all  $1 \leq i \leq n-1$ . Then  $e \in H^x$  if and only if there is a walk  $(x_1, e_1, x_2, e_2, \dots, x_n, e_n)$  from  $x$  to  $e$  with  $x_i \leq x$ , for all  $1 \leq i \leq n$ . Note that  $\{e \in H : x \in e\} \subseteq H^x$ . We define  $H^x[\geq x] := H^x[V(H^x)_{\geq x}] = \{e \cap V(H^x)_{\geq x} : e \in H^x, e \cap V(H^x)_{\geq x} \neq \emptyset\}$ , where  $V(H^x)_{\geq x} = \{y \in V(H^x) : y \geq x\}$ . Observe that  $x \in V(H^x[\geq x])$ . The *coverwidth* of the ordering  $<$  is  $\max_{x \in V(H)} \beta\text{-cn}(H^x[\geq x])$ . The *coverwidth* of  $H$ , denoted by  $\text{cw}(H)$ , is the minimum coverwidth over all orderings of  $V(H)$ . It was shown in [6] that bounded coverwidth implies tractability of Max-CSP:

**Theorem 36** ([6]). *Let  $k \geq 1$  be fixed. There exists an algorithm which, given as input a Max-CSP instance  $I$  with hypergraph  $H$  and an ordering of  $V(H)$  of coverwidth  $\leq k$ , computes  $\text{opt}(I)$  in time polynomial in  $\|I\|$ .*

The main result of this section is the following:

**Proposition 37.** *For every hypergraph  $H$ , we have  $\text{spw}(H) \leq \text{cw}(H)$ .*

*Proof.* Fix an ordering  $<$  of  $V(H)$  of coverwidth  $\leq k$ , where  $k := \text{cw}(H)$ . Let  $x_{\max} := \max_{<}(V(H))$ . We define  $T$  to be the rooted tree with vertex set  $\{t_x : x \in V(H)\}$  and root  $t_{x_{\max}}$  such that  $t_y$  is the parent of  $t_x$  in  $T$  if and only if  $|V(H^x[\geq x])| \geq 2$  and  $y = \min_{<}(V(H^x[\geq x]) \setminus \{x\})$ , or  $|V(H^x[\geq x])| = 1$  and  $y = x_{\max}$ . For  $t_x \in V(T)$ , we define  $B_{t_x} := \{(y, e) : e \in H^x, y \in e, y \geq x\}$ .

We claim that  $(T, (B_t)_{t \in V(T)})$  is a simplified point decomposition of  $H$  of width  $\leq k$ . To see the bound on the width, note that  $H|_{B_{t_x}} = H^x[\geq x]$ . For condition (1) of simplified point decompositions, given  $e \in H$ , we have that  $\{(y, e) : y \in e\} \subseteq B_{t_x}$ , where  $x = \min_{<}(\{y : y \in e\})$ . For condition (2), we need the following claim:

**Claim 7.** *Suppose that  $t_x$  is a descendent of  $t_y$  in  $T$  and  $|V(H^z[\geq z])| \geq 2$ , where  $t_z$  is the only child of  $t_y$  that is ancestor of  $t_x$ . Then  $H^x \subseteq H^y$ .*

*Proof.* We show the claim by induction. For the base case, assume  $t_y$  is the parent of  $t_x$ , and let  $e \in H^x$ . It follows that there is a walk  $\pi_e$  from  $x$  to  $e$  using vertices  $\leq x$ . Since  $|V(H^y[\geq y])| \geq 2$ , we have  $y = \min_{<}(V(H^y[\geq y]) \setminus \{y\})$ . In particular, there is  $f \in H^y$  with  $y \in f$ , and hence a walk  $\pi_f$  from  $x$  to  $f$  using vertices  $\leq x$ . We can concatenate  $y, \pi_f^{-1}$  and  $\pi_e$ , where  $\pi_f^{-1}$  is the reverse sequence of  $\pi_f$ , and obtain a walk from  $y$  to  $e$  using vertices  $\leq y$  (since  $x < y$ ). Hence,  $e \in H^y$ . Now suppose that  $t_x$  is a descendant of  $t_z$  and  $t_y$  is the parent of  $t_z$ , where  $x < z < y$ . Let  $e \in H^x$ . By induction,  $e \in H^z$ . Using the same argument as above, we obtain that  $e \in H^y$ .  $\blacksquare$

Let  $H'$  be a subhypergraph of  $H$ . Suppose that  $x \in V(H'|_{B_{t_y}}) \cap V(H'|_{B_{t_z}})$ , and  $t_w$  is in the unique path from  $t_z$  to  $t_y$  in  $T$ , where  $x, y, z, w \in V(H)$ . Assume first that  $t_z$  is a descendant of  $t_y$ . Since  $x \in V(H'|_{B_{t_z}})$ , there is a point  $(x, e) \in B_{t_z}$  such that  $x \in e$  and  $e \in H'$ . By definition of  $B_{t_z}$ , we have that  $e \in H^z$ . Since  $w \neq x_{\max}$ , we can apply Claim 7 and obtain that  $e \in H^w$ . Since  $w < y \leq x$ , we have  $(x, e) \in B_{t_w}$ . Therefore,  $x \in V(H'|_{B_{t_w}})$ . Suppose now

that  $t_z$  and  $t_y$  are incomparable in  $T$ . Since  $x \in V(H'|_{B_{t_z}})$ , there is  $e \in H^z$  with  $x \in e$  and  $e \in H'$ . Let  $t_r$  be the only child of  $t_x$  that is ancestor of  $t_z$ . Since  $r \neq x_{\max}$ , by Claim 7, we have that  $e \in H^r$ . As  $r < x$ , we have that  $\{r, x\} \subseteq V(H^r[\geq r])$ . We can then apply Claim 7 and deduce that  $e \in H^x$ . In particular,  $x \in V(H'|_{B_{t_x}})$ . Since  $t_z$  and  $t_y$  are descendent of  $t_x$ , we obtain that  $x \in V(H'|_{B_{t_w}})$  by applying the previous case. Hence condition (2) holds.  $\square$

Together with Theorem 21 and Proposition 28, we obtain that  $\text{mimw}(H) \leq 4 \cdot \text{cw}(H)$ , for every hypergraph  $H$ . In particular, we have:

**Corollary 38.** *Every class of hypergraphs of bounded coverwidth also has bounded MIM-width.*

It follows from the proofs of Propositions 37, 30 and 25, that, given a hypergraph  $H$  and an ordering of  $V(H)$  of coverwidth  $\leq k$ , we can compute in time polynomial in  $\|H\|$ , a branch decomposition of  $H$  of MIM-width  $\leq 4k$ . In particular, we obtain Theorem 36 as a consequence of Theorem 19.

Finally, we show that the converse to Corollary 38 does not hold:

**Proposition 39.** *There exists a class of hypergraphs with bounded MIM-width and unbounded coverwidth.*

*Proof.* For every  $n \geq 1$ , we define  $H_n$  to be the hypergraph with vertex set  $X \cup Y$ , where  $X = \{x_1, \dots, x_n\}$  and  $Y = \{y_1, \dots, y_n\}$  and edges  $H = \{X \cup y : y \in Y\} \cup \{Y \cup x : x \in X\}$ . Let  $\mathcal{C} := \{H_n : n \geq 1\}$ . We also define  $e_x := Y \cup x$ , for every  $x \in X$ ; and  $e_y := X \cup y$ , for every  $y \in Y$ .

We first prove that  $\mathcal{C}$  has unbounded coverwidth by showing that  $\text{cw}(H_n) \geq n$ , for every  $n \geq 1$ . Let  $z_1, \dots, z_{2n}$  be any ordering of  $V(H_n)$  and assume without loss of generality that  $z_1 \in X$ . Observe that  $H^{z_1}[\geq z_1] = \{e \in H_n : z_1 \in e\}$ . Then we have  $\{e_{y_1}, \dots, e_{y_n}\} \subseteq H^{z_1}[\geq z_1]$ . Note that  $\{e_{y_1}, y_1\}, \{e_{y_2}, y_2\}, \dots, \{e_{y_n}, y_n\}$  is an induced matching of  $\text{inc}(H^{z_1}[\geq z_1])$ . By Observation 29, we obtain that  $\beta\text{-cn}(H^{z_1}[\geq z_1]) \geq n$ , and hence, the coverwidth of the ordering  $z_1, \dots, z_{2n}$  is  $\geq n$ . Since this holds for any ordering, we have that  $\text{cw}(H_n) \geq n$ .

Now we show that  $\text{mimw}(H_n) \leq 2$ , for every  $n \geq 1$ . We define a branch decomposition  $(T, \delta)$  for  $\text{inc}(H_n)$  as follows. Let  $P$  be the rooted path

$$t_{1,1}, t_{1,2}, \dots, t_{n,1}, t_{n,2}, s_{1,1}, s_{1,2}, \dots, s_{n,1}, s_{n,2}$$

with root  $t_{1,1}$ . The tree  $T$  is obtained from  $P$  by adding, for every  $1 \leq i \leq n$ , fresh nodes  $t'_{i,1}, t'_{i,2}$ , whose parents are  $t_{i,1}, t_{i,2}$ , respectively; and by adding for every  $1 \leq i \leq n-1$ , fresh nodes  $s'_{i,1}, s'_{i,2}$ , whose parents are  $s_{i,1}, s_{i,2}$ , respectively, and a fresh node  $s'_{n,1}$  with parent  $s_{n,1}$ . For every  $1 \leq i \leq n$ , we let  $\delta(t'_{i,1}) = x_i$  and  $\delta(t'_{i,2}) = e_{x_i}$ ; for every  $1 \leq i \leq n-1$ , we let  $\delta(s'_{i,1}) = y_i$  and  $\delta(s'_{i,2}) = e_{y_i}$ ; and we set  $\delta(s'_{n,1}) = y_n$  and  $\delta(s_{n,2}) = e_{y_n}$ .

We claim that the MIM-width of  $(T, \delta)$  is at most 2. Let  $t$  be an internal node (i.e., not a leaf) of  $T$ . Suppose that  $t = t_{i,1}$  for some  $1 \leq i \leq n$  (the case  $t = s_{i,1}$  is analogous). Then we have that  $\text{inc}(H_n)[V_t, V(\text{inc}(H_n)) \setminus V_t]$  is the disjoint union of two complete bipartite graphs: one with partition  $(\{x_1, \dots, x_{i-1}\}, \{e_{y_1}, \dots, e_{y_n}\})$  and the other with partition  $(\{e_{x_1}, \dots, e_{x_{i-1}}\}, \{y_1, \dots, y_n\})$ . In particular,  $\text{MIM}(\text{inc}(H_n)[V_t, V(\text{inc}(H_n)) \setminus V_t]) \leq 2$ . Now suppose that  $t = t_{i,2}$  for some  $1 \leq i \leq n$  (again, the case  $t = s_{i,2}$  is analogous). In this case,  $\text{inc}(H_n)[V_t, V(\text{inc}(H_n)) \setminus V_t]$  is the union of a complete bipartite graph with partition  $(\{e_{x_1}, \dots, e_{x_{i-1}}\}, \{y_1, \dots, y_n\})$  and the graph obtained from the complete bipartite graph with partition  $(\{x_1, \dots, x_i\}, \{e_{y_1}, \dots, e_{y_n}\})$  by adding the vertex  $e_{x_i}$  and the edge  $\{x_i, e_{x_i}\}$ . Hence,  $\text{MIM}(\text{inc}(H_n)[V_t, V(\text{inc}(H_n)) \setminus V_t]) \leq 2$ . We conclude that  $\text{mimw}(H_n) \leq 2$ , and therefore, that  $\mathcal{C}$  has bounded MIM-width.  $\square$