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, β-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 β-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 β-hypertreewidth.

I. Introduction

The *Constraint Satisfaction Problem* (CSP) is a well-known framework for expressing a wide range of both theoretical and real-life combinatorial problems [14], [25], [28]. Some examples are satisfiability [33], evaluation of conjunctive queries [10], [26], graph colorings [23] and homomorphisms [24]. 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.

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 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 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 [16] and [20] (see also [22]) that CSP($\mathcal{H}, -$) is tractable if and only if $\mathcal{H}$ has *bounded treewidth* (under the complexity theoretical assumption that FPT $\neq$ 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. 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 CSP($\mathcal{H}, -$) is tractable. One of the first such classes are the *acyclic* hypergraphs [2], [3], [37] (also called *α-acyclic* [13]). This tractability result has been extended to more general classes such as hypergraphs of bounded *hypertreewidth* [18] and bounded *fractional hypertreewidth* [21]. 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 [27] for a brief discussion on that topic). However, as shown in [27], the classes $\mathcal{H}$ for which 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 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 $f : D^r \to \mathbb{Q}_{\geq 0}$ (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

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.

978-1-7281-3608-0/19/$31.00

[1] A usual definition of a Max-CSP instance is a CSP instance with the goal to maximise the number of satisfied constraints. As we explain in Section II-B, we actually consider a more general framework, sometimes called *finite-valued* CSPs [35] or Max-CSPs with payoff functions [29]. Since our main result is a tractability result, this makes it only stronger.

$X = \{x_1, \ldots, x_n\}$, a domain D of values and a set $\mathcal{C}$ of (finite-valued) constraints, the goal is to compute the maximum value of $f(x_1, \ldots, x_n) = \sum_{f_c(\mathbf{x}) \in \mathcal{C}} f_c(\mathbf{x})$, over all possible assignments of values to X.

In the case of bounded arity, tractability of Max-CSP($\mathcal{H}, -$) is also characterised by bounded treewidth, which follows directly from the CSP case. However, the complexity of unbounded-arity Max-CSPs under structural restrictions is poorly understood and the techniques used in the CSP context cannot be easily applied. Indeed, Max-CSP($\mathcal{H}, -$) is hard even for classes $\mathcal{H}$ of α-acyclic hypergraphs [17]. 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 Max-CSP($\mathcal{H}, -$) are *β-acyclicity*[2] [4], introduced in [13], and having *bounded (incidence) MIM-width*[3] [32], [36]. 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.

A. Contributions

As our main contribution, we introduce the notions of *point decomposition* and *point-width* that unify β-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 4). Our tractability result explains the tractability of β-acyclic and bounded MIM-width hypergraphs. In particular, we prove that every β-acyclic hypergraph has a point decomposition of width 1 and polynomial size (Theorem 7), 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 10), where the latter is defined via a syntactic restriction of point decompositions. Finally, we also discuss some related notions such as β-hypertreewidth [19] (Section VII).

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 III and VI, respectively.

The algorithm behind our main tractability result (Theorem 4) 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 [15]). 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.

B. 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 underlying relational structures follows from [11] and [20]. 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], [38], and (finite-valued) Max-CSPs [35].

C. Structure

Section II introduces the necessary notation on hypergraphs and Max-CSPs. Section III defines point decompositions and point-width. The main tractability result is given in Section IV. Sections V and VI show that β-acyclicity and bounded MIM-width are special cases of bounded point-width, respectively. We conclude in Section VII. All omitted proofs can be found in the full version [9].

II. Preliminaries

A. Hypergraphs, points and covers

We assume that the reader is familiar with elementary graph theory and refer to Diestel's textbook for more details [12]. 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

[2] 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.

[3] The results for MIM-width in [32], [36] apply to Max-SAT (and #SAT), but can be adapted to Max-CSPs. Let us also remark that in [32], [36] 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.

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 $H'|_{P \cap P(H')}$.

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 $\text{cn}(H)$, is the smallest cardinality of an edge cover of H. We denote by $\beta\text{-cn}(H)$ the maximum of $\text{cn}(H')$ over all subhypergraphs H' of H.

B. Max-CSP

A finite-valued *function* of arity $r = \text{ar}(f)$ over a domain D is a mapping $f : D^r \to \mathbb{Q}_{\geq 0}$. 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 $X = \{x_1, \ldots, x_n\}$ of variables, a finite domain D of values, and an objective function of the form

$$f_I(x_1, \ldots, x_n) = \sum_{i=1}^{q} f_i(\mathbf{x_i})$$

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

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

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 [35] or Max-CSPs with payoff functions [29].

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 $\psi : X' \to D$ 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 ψ a *partial assignment to* X. In particular, for any partial assignment ψ to X, we have that $\text{val}(\psi) \leq \text{opt}(I)$. Finally, given a partial assignment $\psi : X' \to D$, we say that ψ *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 ψ can satisfy edges that are not completely contained in X'.

III. 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, $V(H) = \{x_0, x_1, x_2, x_3\}$. 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) 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 1 (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 $(s_1, s_2) \notin E(A)$).

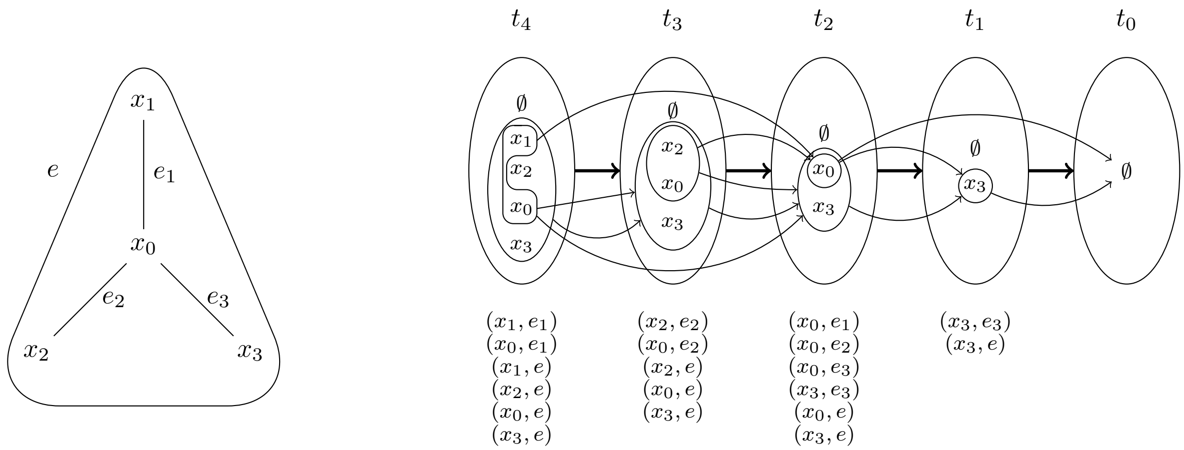

Fig. 1. The hypergraph H and its point decomposition from Examples 1–6.

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).

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 2 (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 3 (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

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

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 (see the full version [9, 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.

IV. 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 4.

Definition 4 (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 1. *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*

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

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 $(s, s_1, \ldots, s_n)$ 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 $V(A') \cup \{s_1, \ldots, s_n\}$ is a realisation and $T_{A'}$ is precisely the subtree of T_{A^*} rooted at s, so the observation follows. □

Definition 5 (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 6 (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 $\psi : S \to D$ such that ψ satisfies some guard C of s. A* consistent assignment *to a partial realisation A' of A is a function ϕ 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 1.

Observation 2. *Let H be the hypergraph of a Max-CSP instance, $(T, (B_t)_{t \in V(T)}, A)$ be a point decomposition of H, ϕ 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 $\psi : X' \to D$ such that for every $s = (t, S) \in V(A')$, $\phi(s) = \psi|_S$.*

Definition 7. *Let H be the hypergraph of a Max-CSP instance, $(T, (B_t)_{t \in V(T)}, A)$ be a point decomposition of H, ϕ be a consistent assignment to a partial realisation A' of A and ψ be as in Observation 2. The* value *of (ϕ, A') is the quantity*

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

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 ψ of the best value achievable by a partial realisation A' with sink s and consistent assignment to A' that agrees with ψ 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.

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 3. *Let H be the hypergraph of a Max-CSP instance, $(T, (B_t)_{t\in V(T)}, A)$ be a point decomposition of H, ϕ be a consistent assignment to a partial realisation A' of A with sink $s = (t, S)$ and ψ be as in Observation 2. 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,* $\mathit{val}(\phi, A') = \sum_{e\in H: e\subseteq S} f_e(\psi(\mathbf{x_e})) +$

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

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 1, if there exist an edge $e \in H$ and two sub-bags $s_1, s_2 \in W$ with $e \subseteq \left(\cup_{(t^*,S^*)\in V(A'[s_1])} S^*\right) \cap \left(\cup_{(t^*,S^*)\in V(A'[s_2])} S^*\right)$ then $e \subseteq S$. Similarly, if there exist $e \in H$ and $s_1 = (t_1, S_1) \in W$ such that $e \subseteq \left(\cup_{(t^*,S^*)\in V(A'[s_1])} S^*\right) \cap S$, then $e \subseteq S_1$. Putting everything together we have

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

as claimed. ☐

Proposition 1. *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 $\mathit{val}(\phi, A')$ over all realisations A' of A and consistent assignments ϕ to A' is exactly $\mathit{opt}(I)$.*

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

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

We now prove $\mathrm{opt}(I) \geq M$. Let A' be a realisation of A and ϕ be a consistent assignment to A' such that $\mathrm{val}(\phi, A') = M$. By Observation 2, there exists an assignment ψ to $X' := \cup_{(t,S)\in V(A')} S$ such that

$$\begin{aligned}
\mathrm{val}(\psi) = \sum_{e\in H: e\subseteq X'} f_e(\psi(\mathbf{x_e})) \;\geq \\
\sum_{e\in H: \exists (t,S)\in V(A'),\, e\subseteq S} f_e(\psi(\mathbf{x_e})) = \mathrm{val}(\phi, A') = M
\end{aligned}$$

and hence $\mathrm{opt}(I) \geq M$. ☐

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

Theorem 4. *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 $\mathit{opt}(I)$ 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 ψ we will associate a nonnegative rational value $\mathrm{val}_{\mathrm{alg}}(s, \psi)$. 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 ψ be an s-valid assignment. Suppose that the values $\mathrm{val}_{\mathrm{alg}}(s', \psi')$ have already been computed for all pairs $(s' = (t', S'), \psi')$ with $t' <_T t$. If t is a leaf then we set $\mathrm{val}_{\mathrm{alg}}(s, \psi) := \sum_{e\in H: e\subseteq S} f_e(\psi(\mathbf{x_e}))$. 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 ψ' 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
$$\mathrm{val}_{\mathrm{alg}}(s', \psi') - \sum_{e\in H: e\subseteq S\cap S'} f_e(\psi(\mathbf{x_e}))$$
over all s'-valid assignments ψ' such that $\psi'|_{S\cap S'} = \psi|_{S\cap S'}$,

and we set $\mathrm{val}_{\mathrm{alg}}(s,\psi) := \sum_{e\in H: e\subseteq S} f_e(\psi(\mathbf{x_e})) + \max_{U\in \mathrm{IS}(G)} \left(\sum_{s'\in U} w(s')\right)$. Once $\mathrm{val}_{\mathrm{alg}}(s,\psi)$ is computed for all pairs (s,ψ) where s is a sub-bag of the root of T, the algorithm outputs the maximum of $\mathrm{val}_{\mathrm{alg}}(s,\psi)$ 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 ψ, $\mathrm{val}_{\mathrm{alg}}(s,\psi)$ is the maximum of $\mathrm{val}(\phi, A')$ over all partial realisations A' of A whose sink is s and consistent assignments ϕ to A' such that $\phi(s) = \psi$.*

We proceed by induction, proving the claim for all pairs (s,ψ) in the same order the algorithm computes $\mathrm{val}_{\mathrm{alg}}(s,\psi)$. Let $s = (t, S)$ be a sub-bag with a directed path in A to a sub-bag of the root of T and ψ be an s-valid assignment. Suppose that the claim holds for all pairs (s',ψ') for which $\mathrm{val}_{\mathrm{alg}}(s',\psi')$ is computed by the algorithm before $\mathrm{val}_{\mathrm{alg}}(s,\psi)$ (and in particular for all pairs (s',ψ') 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 ϕ 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 $T_{A'}$. By definition of $T_{A'}$, W is a subset of $V(G)$ and form an independent set. Furthermore, by Observation 3 and the induction hypothesis we have

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

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

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

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

At this point, we need only prove that there exist a partial realisation A' with sink s and a consistent assignment ϕ to A' such that $\phi(s) = \psi$ and $\mathrm{val}(\phi, A')$ is exactly $\mathrm{val}_{\mathrm{alg}}(s,\psi)$. Let W be the independent set of G chosen by the algorithm to compute $\mathrm{val}_{\mathrm{alg}}(s,\psi)$. For each sub-bag $s' = (t', S') \in W$, let $\psi_{s'}$ be an s'-valid assignment such that $\mathrm{val}_{\mathrm{alg}}(s',\psi_{s'}) - \sum_{e\in H: e\subseteq S\cap S'} f_e(\psi(\mathbf{x_e})) = w(s')$ 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 $\mathrm{val}(\phi_{s'}, A'_{s'}) = \mathrm{val}_{\mathrm{alg}}(s',\psi_{s'}) = w(s') + \sum_{e\in H: e\subseteq S\cap S'} f_e(\psi(\mathbf{x_e}))$. 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 ϕ 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 3 and the induction hypothesis we obtain

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

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

Corollary 1. *The output of the algorithm is the maximum of $\mathrm{val}(\phi, A')$ over all realisations A' of A and consistent assignments ϕ to A'.*

We deduce from Corollary 1 and Proposition 1 that the algorithm correctly outputs $\mathrm{opt}(I)$. 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,ψ), the associated graph G is chordal.*

By way of contradiction let us assume that there exists a pair (s,ψ) 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.

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

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, ψ).

The computation of $\text{val}_{\text{alg}}(s, \psi)$ for a given pair (s, ψ) 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 [15] by Claim 2. Constructing the graph G takes time polynomial in $\|P\|$ and $|H|^k q^k$, which concludes the proof. ☐

V. Relationship with β-acyclicity

A hypergraph H is *α-acyclic* [3] if it has a *join tree*. A join tree is a pair (T, λ) where T is a tree and λ 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 *β-acyclic* [13] if every subhypergraph of H is α-acyclic. It is known that β-acyclic hypergraphs are tractable for Max-CSPs:

Theorem 5 ([4])**.** *Max-CSP($\mathcal{H}, -$) can be solved in polynomial time if $\mathcal{H}$ is a family of β-acyclic hypergraphs.*

The algorithm of Brault-Baron, Capelli, and Mengel [4] works by variable elimination, making use of a well-known alternative characterisation of β-acyclic hypergraphs in terms of the so-called *β-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 4, we can obtain Theorem 5.

An ordering $(x_1, \dots, x_n)$ of the vertices of a hypergraph H is a *β-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 β-acyclic if and only if it has a β-elimination order [3].

Our construction of point decompositions for β-acyclic hypergraphs is inspired by recent work of Capelli [7], from whom we borrow some notation and lemmas. Let H be a β-acyclic hypergraph and $<_\beta$ be a β-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 Δ 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 β-acyclic. A possible β-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 1 ([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 6 ([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 7. *Every β-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 β-acyclic hypergraph with β-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 β-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 β-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 β-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:

- $|S_x| = 1$ 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 $e \in H_f^y$;
- $|S_x| > 1$ 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 $z = \min_{<_\beta}(S_x \backslash \{x\})$.

In addition, if $|S_x| = 1$ we add the arc $((t_x, S_x), (t_\perp, \emptyset))$. By construction, A is a $\mathcal{T}$-structure. The next claim will be used in conjunction with Lemma 1 and Theorem 6 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 $e \in H^y_f$.*

We prove the claim by induction on the length of the path. If the path has length 1, then the claim holds by the definition of A. Now, suppose that the path has length $n > 1$ and that the claim holds for all paths of length $n-1$. Let $z \in V(H)$ be such that $s_z = (t_z, S_z)$ is the predecessor of s_y in the path. Note that S_z cannot be empty. By induction, there exist $e, f \in H$ such that $S_x = e \cap V(H)_{\geq x}$, $S_z = f \cap V(H)_{\geq z}$ and $e \in H^z_f$. Since (s_z, s_y) is an arc in A, there exist $f', g \in H$ such that $S_z = f' \cap V(H)_{\geq z}$, $S_y = g \cap V(H)_{\geq y}$ and $f' \in H^y_g$. If $f <_H g$, then $f \in H^y_g$ as a walk to f' can be extended to a walk to f by going through z. This implies that $e \in H^y_g$ and the claim would follow. If instead $g <_H f$, then by Theorem 6 we have $f \cap V(H)_{\geq y} = f' \cap V(H)_{\geq y} \subseteq V(H^y_g) \cap V(H)_{\geq y} \subseteq g$. It follows from $g <_H f$ that $f \cap V(H)_{\geq y} = g \cap V(H)_{\geq y} = S_y$, and the claim follows since $e \in H^z_f$ implies $e \in H^y_f$.

Claim 5. *A is decomposable.*

We prove the claim by contradiction. Suppose that A is not decomposable, that is, there exist five sub-bags $s, s_x = (t_x, S_x), s_y = (t_y, S_y), s^1_z = (t_z, S^1_z), s^2_z = (t_z, S^2_z)$ 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^1_z to s_x and from s^2_z to s_y. By the definition of A, we can further assume that none of S_x, S_y, S^1_z, S^2_z is empty.

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

We distinguish two cases:

- $f_x \leq_H f_y$. Observe that $z \in e^1_z \cap e^2_z \cap V(H)_{\leq x} \subseteq V(H^x_{f_x}) \cap V(H^y_{f_y}) \cap V(H)_{\leq x}$, so by Lemma 1 we have $H^x_{f_x} \subseteq H^y_{f_y}$. In particular, it holds that $f_x \in H^y_{f_y}$. 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 $w \leq_\beta \min_{<_\beta}(S_x \backslash \{x\})$. 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 $z \in V(H^y_{f_x}) \cap V(H^y_{f_y}) \cap V(H)_{\leq y}$, so by Lemma 1 we have $H^y_{f_y} \subseteq H^y_{f_x}$. By Theorem 6 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 $w \leq_\beta \min_{<_\beta}(S_x \backslash \{x\})$. Again, the arc (s_y, s) implies that $y <_\beta w$. Finally, since $y \in S_x \backslash \{x\}$, we have $w \leq_\beta \min_{<_\beta}(S_x \backslash \{x\}) \leq_\beta y <_\beta w$, a contradiction.

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

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 is the smallest vertex in e with respect to $<_\beta$. Hence condition (i) holds.

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

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

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 $|S_x| > 1$ since otherwise we would have $(s_x, s_\perp)$ as an arc in A'. Let $y = \min_{<_\beta}(S_x \backslash \{x\})$, and let $e_x \in H'$ be such that $S_x = e_x \cap V(H)_{\geq x}$. Let $s_y = (t_y, S_y)$ be the (unique) sub-bag of t_y in $V(A')$, and let $e_y \in H'$ be such that $S_y = e_y \cap V(H)_{\geq y}$. 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 and thus of A', so this cannot be the case. Recall that $<_\beta$ is a β-elimination order, $e_x \in H'$ and $y \in e_x$, so we must have $e_x \cap V(H)_{\geq y} \subset e_y \cap V(H)_{\geq y}$. It follows that $e_x <_H e_y$, and since (e_y, y, e_x) is a walk in H we have $e_x \in H^y_{e_y}$. Hence (s_x, s_y) is an arc in A, a contradiction.

For condition (iii), let A' be a realisation of A and $x \in \cup_{(t,S) \in V(A')} S$. By the definition of A and Theorem 6, for any arc (s, s') of A' where $s = (t_y, S_y)$, $y \in V(H)$ and $s' = (t', S')$ it holds that $S_y \backslash S' = \{y\}$. It follows 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.

The point decomposition $(T, (B_t)_{t \in V(T)}, A)$ has polynomial size. Moreover, it can be computed in polynomial time since a β-elimination order can be computed efficiently from H. 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 7. □

Figure 1 shows the construction from the proof of Theorem 7 applied to the β-acyclic hypergraph H to the left and β-elimination order $x_1 <_\beta x_2 <_\beta x_0 <_\beta x_3$. Note how we need a non-flat point decomposition. It can be verified that the construction produces a non-flat point decomposition independently of the β-elimination order we pick for H. As we shall see in the next section, this is not coincidence as β-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 β-acyclic hypergraphs [4].

VI. Flat point-width and MIM-width

In this section, we show how our main tractability result from Theorem 4 also explains the tractability of Max-CSPs for classes of hypergraphs of bounded MIM-width [32], [36]. 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 $\{u, v\} \not\subseteq \bigcup_{\{u',v'\} \in M} \{u', v'\}$. For a graph G, we denote by $\mathrm{MIM}(G)$ 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, δ) where T is a binary rooted tree and δ 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, δ) is the maximum $\mathrm{MIM}(G[V_t, V(G) \setminus V_t])$, taken over all $t \in V(T)$. The MIM-width [36] of G, denoted by $\mathrm{mimw}(G)$, is the minimum MIM-width over all branch decompositions of G.

The *incidence graph* of a hypergraph H, denoted by $inc(H)$, 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 $\mathrm{mimw}(H)$ of the hypergraph H to be $\mathrm{mimw}(inc(H))$. It follows from the work of Sæther, Telle and Vatshelle [32] 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 8 ([32])**.** *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 $inc(H)$ of MIM-width at most k, computes $opt(I)$ in time polynomial in $\|I\|$.*

Let us stress that the results in [32], [36] are given for Max-SAT (and #SAT). However, Theorem 8 can be obtained by adapting the algorithm from [32], [36] to Max-CSPs. We omit the details as Theorem 8 is implied by the results of this section.

The goal of this section is to prove the following:

Theorem 9. *Let $k \geq 1$ be fixed. For every hypergraph H and branch decomposition of $inc(H)$ 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 8 as a consequence of Theorem 9 and Theorem 4. In order to prove Theorem 9, 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 10. *For every hypergraph H, we have $mimw(H) \leq 4 \cdot fpw(H)$ and $fpw(H) \leq 2 \cdot mimw(H)$. 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 10 directly implies Theorem 9. In order to prove Theorem 10, 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 10 may be of independent interest.

A. 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 $f : 2^{V(G)} \to \mathbb{Q}_{\geq 0}$, 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 [30] 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 $\alpha^2_G : 2^{V(G)} \to \mathbb{Q}_{\geq 0}$ as $\alpha^2_G(X) := \alpha^2(G[X])$, 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 mon-$\alpha^2_G : 2^{V(G)} \to \mathbb{Q}_{\geq 0}$ defined by mon-$\alpha^2_G(X) := \min\{\alpha^2_G(Y) : X \subseteq Y \subseteq V(G)\}$, for every $X \subseteq V(G)$.

Observation 11. *For a graph G, we have the following:*

- *α^2_G is* subadditive*, i.e., $\alpha^2_G(X \cup Y) \leq \alpha^2_G(X) + \alpha^2_G(Y)$, for all $X, Y \subseteq V(G)$.*
- *mon-$\alpha^2_G(X) \leq \alpha^2_G(X)$, for all $X \subseteq V(G)$.*
- *mon-α^2_G is* monotone *(unlike α^2_G), i.e., mon-$\alpha^2_G(X) \leq$ mon-$\alpha^2_G(Y)$, if $X \subseteq Y \subseteq V(G)$.*

We are particularly interested in the notions of α^2_G-width and mon-α^2_G-width for a graph G, which we denote by α^2-w(G) and mon-α^2-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 12. *Let G be a graph. Every induced matching in G is a distance 2-independent set in $L(G)$ and vice versa. In particular, MIM$(G) = \alpha^2(L(G))$.*

Below we show that for bipartite graphs, the MIM-width and the α^2-w (and also mon-α^2-w) of the line graph are equivalent, modulo constant factors. The proof is an adaptation of the classical equivalence between treewidth and *branchwidth* [31]. The proof of the following propositions can be found in the full version of the paper [9, Propositions 23 and 24].

Proposition 2. *For every graph G, we have α^2-w$(L(G)) \leq 2 \cdot$ mimw(G).*

Proposition 3. *For every bipartite graph G, we have mimw$(G) \leq 2 \cdot$ mon-α^2-w$(L(G))$.*

By Propositions 2 and 3, for every bipartite graph G: $\frac{1}{2} \cdot \text{mimw}(G) \leq \text{mon-}\alpha^2\text{-w}(L(G)) \leq \alpha^2\text{-w}(L(G)) \leq 2 \cdot \text{mimw}(G)$.

Remark 1. As in the case of treewidth, the widths α^2-w and mon-α^2-w can be related with other notions such as brambles and games. For instance, α^2-w and mon-α^2-w can be lower bounded by the (natural adaptation of the) *bramble number* [34]. Also, mon-α^2-w can be characterised in terms of the monotone version of the *cops and robber game* [34] (this is the reason why we work explicitly with mon-α^2-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 mon-α^2-w$(X) \leq k$. The minimum k for which the cops can win the game in a monotone way is precisely the mon-α^2-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.

B. Proof of Theorem 10

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

Definition 8 (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 spw(H), is the minimum width over all its simplified point decompositions.

Proposition 4. *For every hypergraph H, we have fpw$(H) =$ spw(H).*

Proof. We start by showing fpw$(H) \leq$ 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 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 $(t_0, S_0), \ldots, (t_n, S_n)$, with $n \geq 2$, such that (i) each $(t_i, S_i) \in V(A')$, (ii) $t_0, \ldots, t_n$ 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 $i \in \{1, \ldots, n\}$, 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 $i \in \{1, \ldots, n-1\}$. 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 fpw$(H) \geq$ 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 8 (simplified point decompositions) holds. ☐

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

Observation 13. *For every hypergraph H, we have β-cn(H) = MIM$(inc(H))$. By Observation 12, we have β-cn$(H) = \alpha^2(pg(H))$.*

Proposition 5. *For every hypergraph H, we have spw$(H) \leq \alpha^2$-w$(pg(H))$ and α^2-w$(pg(H)) \leq 2 \cdot$ spw(H).*

Proof. For spw$(H) \leq \alpha^2$-w$(pg(H))$, let $(T, (B_t)_{t \in V(T)})$ be a tree decomposition of $pg(H)$ of α^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 13, we have β-cn$(H|_{B_t}) = \alpha^2(pg(H|_{B_t})) = \alpha^2(pg(H)[B_t]) = \alpha^2_{pg(H)}(B_t)$, for every $t \in V(T)$. Hence, the width of $(T, (B_t)_{t \in V(T)})$ is k. For condition (1) of Definition 8, let $e \in H$ and note that the set $\{(v,e) \in P(H) : v \in e\}$ forms a clique in $pg(H)$. 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 8 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 α^2-w$(pg(H)) \leq 2 \cdot$ 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 13 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. □

Theorem 10 follows from Propositions 5, 4, 2, and 3. Let us stress that given a branch decomposition (T, δ) 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 2 (and due to Proposition 5), from (T, δ) we can efficiently compute a simplified point decomposition for H of width at most $2k$. Finally, the construction in the proof of Proposition 4 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.

VII. Conclusions

We have introduced a new width that unifies β-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 β-*hypertreewidth* (β-hw) [19], bounded β-*fractional hypertreewidth* (β-fhw), and bounded β-*submodular width* (β-subw). In particular, we have β-subw $\leq$ β-fhw $\leq$ β-hw $\leq$ pw. For precise definitions, see the full version of the paper [9, Appendix A].

We have focused on polynomial-time solvability for Max-CSPs. Regarding *fixed-parameter tractability* (FPT), it is easy to show (see the full version [9, Appendix B]) that Marx's classification of CSPs [27] implies an FPT classification of {0,1}-valued Max-CSPs and the FPT frontier is given by the classes with bounded β-submodular width. This classification implies that for a class of unbounded β-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 β-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, β-hw, β-fhw and β-subw). As a related result, which could be interesting in its own right, we show in the full version [9, Appendix C] that bounded β-fractional hypertreewidth collapses to bounded β-hypertreewidth.

REFERENCES

[1] I. Adler, "Width functions for hypertree decompositions," Ph.D. dissertation, Albert Ludwig University of Freiburg, 2006.

[2] C. Beeri, R. Fagin, D. Maier, A. Mendelzon, J. Ullman, and 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.

[3] C. Beeri, R. Fagin, D. Maier, and M. Yannakakis, "On the desirability of acyclic database schemes," *Journal of the ACM*, vol. 30, no. 3, pp. 479–513, 1983.

[4] J. Brault-Baron, F. Capelli, and 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.

[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.

[6] F. Capelli, "Structural restrictions of CNF-formulas: applications to model counting and knowledge compilation," Ph.D. dissertation, Université Paris Diderot, 2016.

[7] ——, "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.

[8] C. Carbonnel, M. Romero, and 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.

[9] C. Carbonnel, M. Romero, and S. Živný, "Point-width and Max-CSPs," Tech. Rep., April 2019, arXiv:1904.07388 [Online]. Available: http://arxiv.org/abs/1904.07388

[10] A. K. Chandra and 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.

[11] V. Dalmau, P. G. Kolaitis, and 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)*, ser. Lecture Notes in Computer Science, vol. 2470. Springer, 2002, pp. 310–326.

[12] R. Diestel, *Graph Theory*, Fourth ed. Springer, 2010.

[13] R. Fagin, "Degrees of Acyclicity for Hypergraphs and Relational Database Schemes," *Journal of the ACM*, vol. 30, pp. 514–550, 1983.

[14] T. Feder and M. Y. Vardi, "The Computational Structure of Monotone Monadic SNP and Constraint Satisfaction: A Study through Datalog and Group Theory," *SIAM Journal on Computing*, vol. 28, no. 1, pp. 57–104, 1998.

[15] A. Frank, "Some polynomial algorithms for certain graphs and hypergraphs," in *Proceedings of the 5th British Combinatorial Conference, 1975*. Utilitas Mathematica, 1975.

[16] 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.

[17] G. Gottlob, G. Greco, and 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*, ser. Lecture Notes in Computer Science, vol. 5556. Springer, 2009, pp. 16–30.

[18] G. Gottlob, N. Leone, and F. Scarcello, "Hypertree decomposition and tractable queries," *Journal of Computer and System Sciences*, vol. 64, no. 3, pp. 579–627, 2002.

[19] G. Gottlob and R. Pichler, "Hypergraphs in model checking: Acyclicity and hypertree-width versus clique-width," *SIAM J. Comput.*, vol. 33, no. 2, pp. 351–378, 2004.

[20] M. Grohe, "The complexity of homomorphism and constraint satisfaction problems seen from the other side," *Journal of the ACM*, vol. 54, no. 1, pp. 1–24, 2007.

[21] M. Grohe and D. Marx, "Constraint solving via fractional edge covers," *ACM Transactions on Algorithms*, vol. 11, no. 1, pp. 4:1–4:20, 2014.

[22] M. Grohe, T. Schwentick, and 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.

[23] P. Hell and J. Nešetřil, "On the Complexity of H-coloring," *Journal of Combinatorial Theory, Series B*, vol. 48, no. 1, pp. 92–110, 1990.

[24] ——, *Graphs and Homomorphisms*. Oxford University Press, 2004.

[25] P. G. Jeavons, "On the Algebraic Structure of Combinatorial Problems," *Theoretical Computer Science*, vol. 200, no. 1-2, pp. 185–204, 1998.

[26] P. G. Kolaitis and 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.

[27] D. Marx, "Tractable hypergraph properties for constraint satisfaction and conjunctive queries," *Journal of the ACM*, vol. 60, no. 6, 2013, article No. 42.

[28] U. Montanari, "Networks of Constraints: Fundamental properties and applications to picture processing," *Information Sciences*, vol. 7, pp. 95–132, 1974.

[29] 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.

[30] N. Robertson and P. D. Seymour, "Graph minors. III. Planar tree-width," *Journal of Combinatorial Theory, Series B*, vol. 36, no. 1, pp. 49–64, 1984.

[31] ——, "Graph minors. X. Obstructions to tree-decomposition," *Journal of Combinatorial Theory, Series B*, vol. 52, no. 2, pp. 153–190, 1991.

[32] S. H. Sæther, J. A. Telle, and M. Vatshelle, "Solving #SAT and MAXSAT by dynamic programming," *J. Artif. Intell. Res.*, vol. 54, pp. 59–82, 2015.

[33] 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.

[34] P. D. Seymour and R. Thomas, "Graph searching and a min-max theorem for tree-width," *Journal of Combinatorial Theory Series B*, vol. 58, no. 1, pp. 22–33, 1993.

[35] J. Thapper and S. Živný, "The complexity of finite-valued CSPs," *Journal of the ACM*, vol. 63, no. 4, 2016, article No. 37.

[36] M. Vatshelle, "New width parameters of graphs," Ph.D. dissertation, University of Bergen, 2012.

[37] 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.

[38] 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.