Tractable structures for constraint satisfaction with truth tables*

Dániel Marx†

September 22, 2009

Abstract

The way the graph structure of the constraints influences the complexity of constraint satisfaction problems (CSP) is well understood for bounded-arity constraints. The situation is less clear if there is no bound on the arities. In this case the answer depends also on how the constraints are represented in the input. We study this question for the truth table representation of constraints. We introduce a new hypergraph measure *adaptive width* and show that CSP with truth tables is polynomial-time solvable if restricted to a class of hypergraphs with bounded adaptive width. Conversely, assuming a conjecture on the complexity of binary CSP, there is no other polynomial-time solvable case. Finally, we present a class of hypergraphs with bounded adaptive width and unbounded fractional hypertree width.
Keywords: computational complexity, constraint satisfaction, treewidth, adaptive width.

1 Introduction

Constraint satisfaction is a general framework that includes many standard algorithmic problems such as satisfiability, graph coloring, database queries, etc. A constraint satisfaction problem (CSP) consists of a set V of variables, a domain D, and a set C of constraints, where each constraint is a relation on a subset of the variables. The task is to assign a value from D to each variable in such a way that every constraint is satisfied (see Definition 4 for the formal definition). For example, 3SAT can be interpreted as a CSP problem where the domain is $D = \{0, 1\}$ and the constraints in C correspond to the clauses (thus the arity of each constraint is 3). For more background, see e.g., [12, 7].

In general, solving constraint satisfaction problems is NP-hard if there are no additional restrictions on the instances. The main goal of the research on CSP is to identify tractable special cases of the general problem. The theoretical literature on CSP investigates two main types of restrictions. The first type is to restrict the *constraint language,* that is, the type of constraints that are allowed. This direction includes the classical work of Schaefer [22] and its many generalizations [3, 2, 4, 7, 17]. The second type is to restrict the *structure* induced by the constraints on the variables. The *hypergraph* of a CSP instance is defined to be a hypergraph on the variables of the instance such that for each constraint $c \in C$ there is a hyperedge E_c that contains all the variables that appear in c. If the hypergraph of the CSP instance has very simple structure, then the instance is easy to solve. For example, it is well-known that a CSP instance I with hypergraph H can be solved in time $\|I\|^{O(\mathrm{tw}(H))}$ [9], where $\mathrm{tw}(H)$ denotes the treewidth of H and $\|I\|$ is the size of the representation of I in the input. Thus if we restrict the problem to instances where the

*Research supported by the Magyary Zoltán Felsőoktatási Közalapítvány, Hungarian National Research Fund (OTKA 67651), and ERC Advanced Grant DMMCA.

†Tel Aviv University, Tel Aviv, Israel, dmarx@cs.bme.hu

treewidth of the hypergraph is bounded by some constant w, then the problem is polynomial-time solvable. The aim of this paper is to investigate whether there exists some other structural property of the hypergraph besides bounded treewidth that makes the problem tractable. Formally, for a class $\mathcal{H}$ of hypergraphs, let $\mathrm{CSP}(\mathcal{H})$ be the restriction of CSP where the hypergraph of the instance is assumed to be in $\mathcal{H}$. Our goal is to characterize the complexity of $\mathrm{CSP}(\mathcal{H})$ for every class $\mathcal{H}$.

We investigate two notions of tractability. $\mathrm{CSP}(\mathcal{H})$ is *polynomial-time solvable* if there is an algorithm solving every instance of $\mathrm{CSP}(\mathcal{H})$ in time $(\|I\|)^{O(1)}$, where $\|I\|$ is the length of the representation of I in the input. The following notion interprets tractability in a less restrictive way: $\mathrm{CSP}(\mathcal{H})$ is *fixed-parameter tractable (FPT)* if there is an algorithm solving every instance I of $\mathrm{CSP}(\mathcal{H})$ in time $f(H)(\|I\|)^{O(1)}$, where f is an arbitrary function and H is the hypergraph of the instance. Equivalently, the factor $f(H)$ in the definition can be replaced with a factor $f(k)$ depending only on the number k of vertices of H: as the number of hypergraphs on k vertices (without parallel edges) is bounded by a function of k, the two definitions result in the same notion. The motivation behind the definition of fixed-parameter tractability is that in certain applications we expect the domain size to be much larger than the number of variables, hence a constant factor in the running time depending only on the number of variables (or on the hypergraph) is acceptable. For example, in the theory of database queries, we can assume that the query size (i.e., the size of the hypergraph) is small, while the database relations are large (see [10, 15]). For a more general treatment of fixed-parameter tractability, the reader is referred to the parameterized complexity literature [6, 8].

Bounded arities. If the constraints have bounded arity (i.e., the edge size in $\mathcal{H}$ is bounded by a constant), then the complexity of $\mathrm{CSP}(\mathcal{H})$ is well understood. In this case, bounded treewidth is the only polynomial-time solvable case:

Theorem 1 ([13]). *If $\mathcal{H}$ is a recursively enumerable class of hypergraphs with bounded edge size, then (assuming* FPT $\neq$ W[1]*) the following are equivalent:*

1. $\mathrm{CSP}(\mathcal{H})$ *is polynomial-time solvable.*
2. $\mathrm{CSP}(\mathcal{H})$ *is fixed-parameter tractable.*
3. $\mathcal{H}$ *has bounded treewidth.*

The assumption FPT $\neq$ W[1] is a standard hypothesis of parameterized complexity. Thus in the bounded arity case bounded treewidth is the only property of the hypergraph that can make the problem polynomial-time solvable. Furthermore, the following sharpening of Theorem 1 shows that there is no algorithm whose running time is significantly better than the $\|I\|^{O(\mathrm{tw}(H))}$ bound of the treewidth based algorithm. The result is proved under the Exponential Time Hypothesis (ETH) [16], a somewhat stronger assumption than FPT $\neq$ W[1]: it is assumed that there is no $2^{o(n)}$ time algorithm for n-variable 3SAT.

Theorem 2 ([19]). *If there is a computable function f and a recursively enumerable class $\mathcal{H}$ of hypergraphs with bounded edge size and unbounded treewidth such that the problem* $\mathrm{CSP}(\mathcal{H})$ *can be solved in time $f(H)\|I\|^{o(\mathrm{tw}(H)/\log \mathrm{tw}(H))}$ for instances I with hypergraph $H \in \mathcal{H}$, then ETH fails.*

This means that the treewidth-based algorithm is almost optimal: in the exponent only an $O(\log \mathrm{tw}(H))$ factor improvement is possible. It is conjectured in [19] that Theorem 2 can be made tight, i.e., the lower bound holds even if the logarithmic factor is removed from the exponent.

Conjecture 3 ([19]). *If $\mathcal{H}$ is a class of hypergraphs with bounded edge size, then there is no algorithm that solves* $\mathrm{CSP}(\mathcal{H})$ *in time $f(H)\|I\|^{o(\mathrm{tw}(H))}$ for instances I with hypergraph $H \in \mathcal{H}$, where f is an arbitrary computable function.*

Unbounded arities. The situation is less understood in the unbounded arity case, i.e., when there is no bound on the maximum edge size in $\mathcal{H}$. First, the complexity in the unbounded-arity case depends on how the constraints are represented. In the bounded-arity case, if each constraint contains at most r variables (r being a fixed constant), then every reasonable representation of a constraint has size $|D|^{O(r)}$. Therefore, the size of the different representations can differ only by a polynomial factor. On the other hand, if there is no bound on the arity, then there can be exponential difference between the size of succinct representations (e.g., formulas) and verbose representations (e.g., truth tables). The running time of an algorithm is expressed as a function of the input size, hence the complexity of the problem can depend on how the input is represented: longer representation means that it is potentially easier to obtain a polynomial-time algorithm.

The most well-studied representation of constraints is listing all the tuples that satisfy the constraint. For this representation, there are classes $\mathcal{H}$ with unbounded treewidth such that CSP restricted to this class is polynomial-time solvable. For example, classes with bounded *(generalized) hypertree width* [11], bounded *fractional edge cover number* [14], and bounded *fractional hypertree width* [14, 20] are such classes. However, no classification theorem similar to Theorem 1 is known for this version. More succinct representations were studied by Chen and Grohe [5]: constraints are represented by generalized DNF formulas or by decision diagrams. The complexity of the problem with these representations were fully characterized: it turns out that the complexity depends not on the treewidth of the hypergraph (as in Theorem 1) but on the treewidth of the incidence structure (in the case of generalized DNF representation) or on the treewidth of a structure describing the decision diagrams (in the case of decision diagram representation).

Truth table representation. In this paper we study another natural representation: truth tables. A constraint of arity r is represented by having one bit for each possible r-tuple that can appear on the r variables of the constraint, and this bit determines whether this particular r-tuple satisfies the constraint or not. This means that the representation of an r-ary constraint consists of $|D|^r$ bits, if the domain of every variable is D. To increase the flexibility of the representation and make it more natural, we allow that the variables have different domains, i.e., each variable v has to be assigned a value from its domain $\mathrm{Dom}(v)$. Thus the size of the truth table of an r-ary constraint is proportional to the size of the direct product of the domains of the r variables. This representation is more verbose than listing satisfying tuples: the size of the representation is proportional to the number of possible tuples even if only few tuples satisfy the constraint. While the motivation for truth table representation is not as strong as for representation by listing all the tuples (which is the natural representation in database-theoretic applications [18, 21]), we believe that investigating truth-table representation is an important theoretical problem and the ideas discovered in this paper will be useful in the study of more natural representations. In particular, the main algorithmic message of the paper ("the decomposition should depend not only on the hypergraph, but also on other properties of the instance") might be relevant in other contexts as well. Furthermore, any hardness result obtained for this representation immediately proves hardness for every more succinct representation.

Formally, we define the variant of CSP considered in this paper as follows:

Definition 4. A *CSP instance* is a quadruple (V, D, Dom, C), where:

- V is a set of variables,
- D is a domain of values,
- $\mathrm{Dom} : V \to 2^D$ assigns a domain $\mathrm{Dom}(v) \subseteq D$ to each variable $v \in V$,
- C is a set of constraints. Each $c_i \in C$ is a pair $\langle s_i, R_i \rangle$, where:

– $s_i = (u_{i,1}, \ldots, u_{i,m_i})$ is a tuple of variables (the *constraint scope*), and
– R_i is a subset of $\prod_{j=1}^{m_i} \mathrm{Dom}(u_{i,j})$ (the *constraint relation*).

For each constraint $\langle s_i, R_i \rangle$ the tuples of R_i indicate the allowed combinations of values for the variables in s_i. The length m_i of the tuple s_i is called the *arity* of the constraint. A *solution* to a CSP instance is a function $f : V \to D$ such that $f(v) \in \mathrm{Dom}(v)$ for every $v \in V$ and for each constraint $\langle s_i, R_i \rangle$ with $s_i = \langle u_{i,1}, u_{i,2}, \ldots, u_{i,m_i} \rangle$, the tuple $\langle f(u_{i,1}), f(u_{i,2}), \ldots, f(u_{i,m_i}) \rangle$ is in R_i.

We denote by $\mathrm{CSP}_{\mathrm{tt}}$ the problem where each constraint $\langle s_i, R_i \rangle$ of arity m_i is represented by the truth table of the constraint relation R_i, that is, by a sequence of $\prod_{j=1}^{m_i} |\mathrm{Dom}(u_{i,j})|$ bits that describe this subset R_i of $\prod_{j=1}^{m_i} \mathrm{Dom}(u_{i,j})$. For a class $\mathcal{H}$, $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ denotes the problem with truth table representation, restricted to instances whose hypergraphs are in $\mathcal{H}$.

Results. The main result of the paper is a complete characterization of the complexity of $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ (assuming Conjecture 3). We introduce a new hypergraph measure *adaptive width*; it turns out the complexity of the problem depends on whether adaptive width is bounded:

Theorem 5 (Main). *Assuming Conjecture 3, the following are equivalent:*

1. $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ *is polynomial-time solvable.*
2. $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ *is fixed-parameter tractable.*
3. $\mathcal{H}$ *has bounded adaptive width.*

The assumption in Theorem 5 is nonstandard, so it is up to the reader to decide how strong this evidence is. However, the message of Theorem 5 is the following: a new tractable class for $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ would imply surprising new results for *binary* CSP. Thus at this point it is not worth putting too much effort in further studying $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ with the hope of finding new tractable classes: as this would disprove Conjecture 3, such an effort would be better spent trying to disprove Conjecture 3 directly, by beating the $\|I\|^{O(\mathrm{tw}(H))}$ algorithm for binary CSP.

Listing the satisfying tuples is a more succinct representation of a constraint than a truth table. Thus if CSP is polynomial-time solvable or fixed-parameter tractable for some class $\mathcal{H}$ with the former representation, then this also holds for the latter representation as well. In particular, this means that by the results of [14, 20], $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ is polynomial-time solvable if $\mathcal{H}$ has bounded fractional hypertree width. This raises the question whether Theorem 5 gives any new tractable class $\mathcal{H}$. In other words, is there a class $\mathcal{H}$ having bounded adaptive width but unbounded fractional hypertree width? In Section 5, we answer this question by constructing such a class $\mathcal{H}$. This means that $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ is polynomial-time solvable, but if the constraints are represented by listing the satisfying tuples, then it is not even known whether the problem is FPT.

2 Width parameters

Treewidth and various variants are defined in this section. We follow the framework of width functions introduced by Adler [1]. A *tree decomposition* of a hypergraph H is a tuple $(T, (B_t)_{t \in V(T)})$, where T is a tree and $(B_t)_{t \in V(T)}$ is a family of subsets of $V(H)$ such that for each $E \in E(H)$ there is a node $t \in V(T)$ such that $E \subseteq B_t$, and for each $v \in V(H)$ the set $\{t \in V(T) \mid v \in B_t\}$ is connected in T. The sets B_t are called the *bags* of the decomposition. Let $f : 2^{V(H)} \to \mathbb{R}^+$ be a function that assigns a nonnegative real number to each nonempty subset of vertices. The *f-width* of a tree-decomposition $(T, (B_t)_{t \in V(T)})$ is $\max\{f(B_t) \mid t \in V(T)\}$. The *$f$-width* of a hypergraph H is the minimum of the f-widths of all its tree decompositions. Now treewidth can be defined as follows:

Definition 6. Let $s(B) = |B| - 1$. The *treewidth* of H is $\mathrm{tw}(H) := s\text{-width}(H)$.

Further width notions defined in the literature can also be conveniently defined using this setup. A subset $E' \subseteq E(H)$ is an *edge cover* if $\bigcup E' = V(H)$. The *edge cover number* $\rho(H)$ is the size of the smallest edge cover (here we assume that H has no isolated vertices). For $X \subseteq V(H)$, let $\rho_H(X)$ be the size of the smallest set of edges covering X.

Definition 7. The *(generalized) hypertree width* of H is $\mathrm{hw}(H) := \rho_H\text{-width}(H)$.

We also consider the linear relaxations of edge covers: a function $\gamma : E(H) \to [0,1]$ is a *fractional edge cover* of H if $\sum_{E:v\in E} \gamma(E) \geq 1$ for every $v \in V(H)$. The *fractional cover number* $\rho^*(H)$ of H is the minimum of $\sum_{E\in E(H)} \gamma(E)$ taken over all fractional edge covers of H. We define $\rho^*_H(X)$ analogously to $\rho_H(X)$: the requirement $\sum_{E:v\in E} \gamma(E) \geq 1$ is restricted to vertices $v \in X$.

Definition 8. The *fractional hypertree width* of H is $\mathrm{fhw}(H) := \rho^*_H\text{-width}(H)$.

The dual of covering is independence. A subset $X \subseteq V(H)$ is an *independent set* if $|X \cap E| \leq 1$ for every $E \in E(H)$. The *independence number* $\alpha(H)$ is the size of the largest independent set and $\alpha_H(X)$ is the size of the largest independent set that is a subset of X. A function $\phi : V(H) \to [0,1]$ is a *fractional independent set* of the hypergraph H if $\sum_{v\in E} \phi(v) \leq 1$ for every $E \in E(H)$. The *fractional independence number* $\alpha^*(H)$ of H is the maximum of $\sum_{v\in V(H)} \phi(v)$ taken over all fractional independent sets ϕ of H and $\alpha^*_H(X)$ is the maximum taken over all independent sets that are zero outside X. It is well known that $\alpha(H) \leq \alpha^*(H) = \rho^*(H) \leq \rho(H)$ for every hypergraph H and hence $\alpha^*_H(X) = \rho^*_H(X)$ for every $X \subseteq V(H)$. Thus α^*_H-width gives us exactly the same notion as fractional hypertree width. The main new definition of the paper uses fractional independent sets, but in a different way. For a function $f : V(H) \to \mathbb{R}^+$, we define $f(X) = \sum_{v\in X} f(v)$ for $X \subseteq V(H)$ and define f-width accordingly.

Definition 9. The *adaptive width* $\mathrm{adw}(H)$ of a hypergraph H is the maximum of $\phi\text{-width}(H)$ taken over all fractional independent sets ϕ of H.

The difference between fractional hypertree width and adaptive width can be understood the following way. As mentioned above, in Definition 8, ρ^*_H-width can be replaced by α^*_H-width, i.e., we are interested in a tree decomposition where the size of the maximum fractional independent set is bounded in every bag. In other words,

$$\begin{gathered}\mathrm{fhw}(H) \leq w \\ \Updownarrow \\ \textit{exists} \text{ a tree decomposition } \mathcal{T} \text{ such that} \\ \textit{for all} \text{ fractional independent set } \phi, \\ \phi(B) \leq w \text{ for every bag } B \text{ of } \mathcal{T}.\end{gathered}$$

The definition of adaptive width exchanges the two quantifiers. Instead of requiring that there is a tree decomposition that is "good" for every fractional independent set ϕ, we require that for every fractional independent set ϕ, there is a tree decomposition that is "good." More precisely,

$$\begin{gathered}\mathrm{adw}(H) \leq w \\ \Updownarrow \\ \textit{for all} \text{ fractional independent set } \phi, \\ \textit{exists} \text{ a tree decomposition } \mathcal{T} \text{ such that} \\ \phi(B) \leq w \text{ for every bag } B \text{ of } \mathcal{T}.\end{gathered}$$

For constraint satisfaction problems, bounded fractional hypertree width means that for every hypergraph H, there is a tree decomposition such that every instance has a bounded number of solutions in each bag, thus we can use dynamic programming to solve the instance [14]. The main conceptual contribution of this paper is the idea that we should look at the instance first and *use different tree decompositions for different instances.* In the truth table setting, we look at the distribution of the domain sizes in the input instance and derive a fractional independent set ϕ based on these sizes. Bounded adaptive width means that there is a tree decomposition where this particular ϕ is bounded on each bag, and, as we shall see in Section 3, this implies that there is only a bounded number of solutions in each bag. Thus we are not using a single fixed decomposition for each hypergraph, but we find the decomposition *adaptively,* taking into account the properties of the actual instance. This paradigm (finding a decomposition *after* looking at the instance) is the main message of the paper and might be of use in other settings as well.

Currently, we do not have an efficient algorithm for computing adaptive width. Fortunately, the polynomial-time algorithm in Section 3 for instances with bounded adaptive width does not need to determine the adaptive width of the input, it is sufficient that the adaptive width is promised to be bounded. However, the technical details of the hardness proof of Section 4 require that the question $\mathrm{adw}(H) \geq w$ is decidable. The following lemma gives an algorithm for this decision problem.

Lemma 10. *There is an algorithm that, given a hypergraph H and a rational number w, decides if $\mathrm{adw}(H) \geq w$. If the answer is yes, then the algorithm returns a rational fractional independent set α such that the α-width of H is at least w.*

Proof. The hypergraph H has a finite number d of tree decompositions, let us fix an enumeration of these decompositions. Let $\mathcal{B}_i$ be the set of bags of the i-th decomposition. If $\mathrm{adw}(H) \geq w$, then there is a fractional independent set ϕ for H such that the ϕ-width of H is at least w. This means that for every $1 \leq i \leq d$, there is a bag $B_i \in \mathcal{B}_i$ such that $\phi(B_i) \geq w$. Conversely, if ϕ is a fractional independent set of H such that for every $1 \leq i \leq d$, there is a bag $B_i \in \mathcal{B}_i$ with $\phi(B_i) \geq w$, then surely $\mathrm{adw}(H) \geq w$. Our algorithm tries every possible vector $(B_1, \ldots, B_d)$ satisfying $B_i \in \mathcal{B}_i$ for every $1 \leq i \leq d$, and checks whether there is a suitable fractional independent set ϕ. Clearly, there is only a finite number of such vectors. If there is a suitable ϕ for at least one vector, then $\mathrm{adw}(H) \geq w$ follows; if there is no such ϕ for any vector, then $\mathrm{adw}(H) < w$.

For a given vector $(B_1, \ldots, B_d)$, we have to check whether there exists an $\phi : V(H) \to \mathbb{R}^+$ such that

$$\phi(E) \leq 1 \text{ for every } E \in E(H),$$
$$\phi(B_i) \geq w \text{ for every } 1 \leq i \leq d$$

holds. This can be decided with the use of linear programming. If there is a suitable ϕ, then it follows from well-known results in linear programming that there is a rational ϕ as well. □

We finish the section with a combinatorial observation that will be useful later (recall that the closed neighborhood of a vertex v is the union of all the edges containing v):

Lemma 11. *Given a tree decomposition of hypergraph H, it can be transformed in polynomial time into a tree decomposition satisfying the following property: if two adjacent vertices u and v have the same closed neighborhood, then u and v appear in exactly the same bags. Furthermore, every bag of the resulting decomposition is a subset of a bag of the original decomposition.*

Proof. Consider a tree decomposition of H. If u and v are two vertices that do not satisfy the requirements, then remove these vertices from those bags where only one of them appears (since u

and v are neighbors, they appear together in some bag B_t, hence both vertices appear in at least one bag after the removals). The intersection of two subtrees is also a subtree, thus it remains true that u and v appear in a connected subset of the bags. We have to show that for every edge $E \in E(H)$, there is a bag B_t that fully contains E even after the removals. If $\{u, v\} \subseteq E$ or $E \cap \{u, v\} = \emptyset$, then this clearly follows from that fact that some bag fully contains E before the removals. Assume without loss of generality that $u \in E$ and $v \notin E$. We show that $E \cup \{v\}$ is fully contained in some bag B_t before the removals, hence (as $\{u, v\} \subseteq E \cup \{v\}$) B_t fully contains $E \cup \{v\}$ even after the removals. Since $u \in E$, edge E is in the closed neighborhood of u. Thus by assumption, E is also in the closed neighborhood of v, which means that $E \cup \{v\}$ is a clique in H (recall that a clique in hypergraph is set K of vertices such that for any two $x, y \in K$, there is an edge containing both x and y). It is well known that every clique is fully contained in some bag of the tree decomposition (this follows from the fact that subtrees of a tree satisfy the Helly property), thus it follows that $E \cup \{v\} \subseteq B_t$ for some bag B_t.

Let us repeat these removals until there are no pairs u, v that violate the requirements; eventually we get a tree decomposition as required. Observe that the procedure terminates after a polynomial number of steps: vertices are only removed from the bags. □

3 Algorithm for bounded adaptive width

We prove that $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ is polynomial-time solvable if $\mathcal{H}$ has bounded adaptive width. Bounded adaptive width ensures that no matter what the distribution of the domain sizes in the input instance is, there is a decomposition where the variables in each bag have only a polynomial number of possible assignments. For such a decomposition, the instance can be solved by standard techniques.

Lemma 12. *There is an algorithm that, given an instance I of $\mathrm{CSP}_{\mathrm{tt}}$, an integer C, and a tree decomposition $(T, (B_t)_{t \in V(T)})$ of the hypergraph H of the instance such that $\prod_{v \in B_t} |\mathrm{Dom}(v)| \leq C$ for every bag B_t, solves the instance I in time polynomial in $\|I\| \cdot C$.*

Proof. If $\prod_{v \in B_t} |\mathrm{Dom}(v)| \leq C$, then there are at most C possible assignments on the variables in B_t. Using standard dynamic programming techniques, it is easy to check whether it is possible to select one assignment f_t for each bag B_t such that f_t satisfies the instance restricted to the bag B_t and these assignments are compatible. For completeness, we briefly describe how this can be done by a reduction to binary CSP.

Let us construct a binary CSP instance I' as follows. The set of variables of I' is $V(T)$, i.e., the bags of the tree decomposition. For $t \in V(T)$, let $b_t \leq C$ be the number of assignments f to the variables in B_t such that $f(v) \in \mathrm{Dom}(v)$ for every $v \in B_t$; denote by $f_{t,i}$ the i-th such assignment on B_t ($1 \leq i \leq b_t$). The domain of I' is $D' = \{1, \ldots, C\}$. For each edge $t't'' \in E(T)$, we introduce a constraint $c_{t',t''} = \langle (t', t''), R_{t',t''} \rangle$, where $(i, j) \in R_{t',t''}$ if and only if

- $i \leq b_{t'}$ and $j \leq b_{t''}$,
- $f_{t',i}$ and $f_{t'',j}$ are compatible, i.e., $f_{t',i}(v) = f_{t'',j}(v)$ for every $v \in B_{t'} \cap B_{t''}$.
- $f_{t',i}$ satisfies every constraint of I whose scope is contained in $B_{t'}$.
- $f_{t'',j}$ satisfies every constraint of I whose scope is contained in $B_{t''}$.

It is easy to see that a solution of I' determines a solution of I. The size of I' is polynomial in C and $\|I\|$. Since the graph of I' is a tree, it can be solved in time $\|I'\|^{O(1)} = (\|I\|C)^{O(1)}$. □

Theorem 13. *If $\mathcal{H}$ has bounded adaptive width, then $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ is polynomial-time solvable.*

Proof. Let I be an instance of $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ with hypergraph H such that $\mathrm{adw}(H) \leq c$. Let $N \leq \|I\|$ be the size of the largest truth table in the input; we assume that $N > 1$, since the problem is trivial if $N = 1$. We show that it is possible to find in time $N^{O(c)}$ a tree decomposition $(T, B_{t\in V(T)})$ of the instance such that $\prod_{v\in B_t} |\operatorname{Dom}(v)| \leq N^{O(c)}$ holds for every bag B_t. By Lemma 12, this means that the instance can be solved in time polynomial in $\|I\|$ and $N^{O(c)}$, i.e., the running time is $\|I\|^{O(c)}$.

Let $\phi(v) = \log_2 |\operatorname{Dom}(v)| / \log_2 N$. We claim that ϕ is a fractional independent set of H. Indeed, if there is a constraint with $(v_{i_1}, v_{i_2}, \ldots, v_{i_r})$ such that $\sum_{j=1}^{r} \phi(v_j) > 1$, then the size of the truth table describing the constraint is larger than N:

$$\prod_{j=1}^{r} |\operatorname{Dom}(i_j)| = \prod_{j=1}^{r} 2^{\phi(v_{i_j})\cdot\log_2 N} = 2^{\log_2 N \cdot \sum_{j=1}^{r} \phi(v_{i_j})} > 2^{\log_2 N} = N.$$

Define $\phi'(v) = \lceil \phi(v) \log_2 N \rceil$. Observe that $\phi(v) \geq 1/\log_2 N$, hence $\phi'(v) < 2\phi(v)\log_2 N$ (if $|\operatorname{Dom}(v)| = 1$, then the instance can be simplified). Let H' be the hypergraph that is obtained from H by replacing each vertex v with a set X_v of $\phi'(v)$ vertices; if an edge E contains some vertex v in H, then E contains every vertex of X_v in H'. We claim that H' has treewidth less than $2c\log_2 N$. Since $\mathrm{adw}(H) \leq c$, H has a tree decomposition $(T, B_{t\in V(T)})$ such that $\sum_{v\in B_t} \phi(v) \leq c$ holds for every bag B_t. Consider the analogous decomposition $(T, B'_{t\in V(T)})$ of H'; i.e., if a bag B_t contains a vertex v of H, then let bag B'_t contain every vertex of X_v. The size of a bag B'_t is $\sum_{v\in B_t} |X_v| = \sum_{v\in B_t} \phi'(v) \leq 2\log_2 N \cdot \sum_{v\in B_t} \phi(v) \leq 2c\log_2 N$, thus the treewidth of H' is indeed less than $2c\log_2 N$. Given a graph G with n vertices, it is possible to find a tree decomposition of width at most $4\,\mathrm{tw}(G) + 1$ in time $2^{O(\mathrm{tw}(G))}n^{O(1)}$ (see e.g., [8, Prop. 11.14]). Thus we can a find a tree decomposition $(T, B''_{t\in V(T)})$ of width at most $8c\log_2 N$ for H' in time $2^{O(2c\log_2 N)}||H'||^{O(1)} = N^{O(c)}||H'||^{O(1)}$.

In H', every vertex of X_v is contained in the same set of edges. Therefore, by Lemma 11, it can be assumed that each bag of the tree decomposition $(T, B''_{t\in V(T)})$ contains either all or none of X_v. Define the tree decomposition $(T, B^*_{t\in V(T)})$ of H where bag B^*_t contains v if and only X_v is contained in B''_t (it is easy to verify that this is indeed a tree decomposition of H). The ϕ-weight of a bag B^*_t can be bounded as

$$\sum_{v\in B^*_t} \phi(v) \leq \frac{1}{\log_2 N} \sum_{v\in B^*_t} \phi'(v) = \frac{1}{\log_2 N} |B''_t| \leq 8c.$$

Thus in the tree decomposition $(T, B^*_{t\in V(T)})$, the product of the domain sizes is

$$\prod_{v\in B^*_t} |\operatorname{Dom}(v)| = \prod_{v\in B^*_t} 2^{\phi(v)\cdot\log_2 N} = 2^{\log_2 N \cdot \sum_{v\in B^*_t} \phi(v)} \leq 2^{\log_2 N \cdot 8c} = N^{8c},$$

in each bag B^*_t, as required. □

4 Hardness result for unbounded adaptive width

We prove the main complexity result of the paper in this section. The main argument is the following. Suppose that $\mathcal{H}$ is class of hypergraph with unbounded adaptive width such that $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ is

fixed-parameter tractable. Let $H \in \mathcal{H}$ be a hypergraph with $\mathrm{adw}(H) \geq k$ and let ϕ be a fractional independent set such that the ϕ-width of H is at least k. Let us construct a graph G from the graph underlying H by replacing each vertex v with a clique of size $\phi(v) \cdot q$, where q is an appropriate constant. It is not difficult to show that the treewidth of G is roughly qk. In a natural way, any binary CSP instance I_1 whose primal graph is G and whose domain is D can be simulated by a CSP instance I_2 whose hypergraph is H: we set the domain of variable v of I_2 to be $(\phi(v) \cdot q)$-tuples of D, that is, variable v of I_2 simulates $\phi(v) \cdot q$ variables of I_1. If we estimate the cost of solving I_1 by first transforming it to I_2 and then solving it by the assumed algorithm for $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$, it turns out that, compared to solving I_1 directly by using the treewidth-based algorithm, we gain a factor of k in the exponent. Since adaptive width can be arbitrarily large for hypergraphs in $\mathcal{H}$, this gain in the exponent can be arbitrarily large for graphs G arising this way. Thus Conjecture 3 does not hold for the the class $\mathcal{G}$ of graphs constructed from the hypergraphs in $\mathcal{H}$.

Theorem 14. *Let $\mathcal{H}$ be a recursively enumerable class of hypergraphs with unbounded adaptive width. Assuming Conjecture 3,* $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ *is not fixed-parameter tractable.*

Proof. Suppose that $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ can be solved in time $h_1(H)\|I\|^c$ for some constant c and computable function h_1. Let us fix an arbitrary computable enumeration of the hypergraphs in $\mathcal{H}$. For every $k \geq 1$, let H_k be the first hypergraph in this enumeration with $\mathrm{adw}(H_k) \geq k$; as $\mathcal{H}$ has unbounded adaptive width, there is such a graph for every k. For each $k \geq 1$, let ϕ_k be the fractional independent set returned by the algorithm of Lemma 10 for the question '$\mathrm{adw}(H_k) \geq k$?'. Clearly, the ϕ_k-width of H_k is at least k.

Constructing the graph class $\mathcal{G}$. For each $k \geq 1$, we construct a graph G_k based on H_k and ϕ_k. Let q_k be the least common denominator of the rational values $\phi_k(v)$ for $v \in V(H_k)$. The graph G_k has a clique K_v of size $q_k \cdot \phi(v)$ for each $v \in V(H_k)$ and if u and v are neighbors in H_k, then every vertex of K_u is adjacent to every vertex of K_v. Let $\mathcal{G} = \{G_k \mid k \geq 1\}$.

We claim that $\mathrm{tw}(G_k) \geq q_k k - 1$. Suppose for contradiction that G_k has a tree decomposition $(T, (B_t)_{t \in V(T)})$ of width less than $q_k k - 1$, i.e., the size of every bag is smaller than $q_k k$. By Lemma 11, it can be assumed that for every $v \in V(H_k)$ and bag B_t of the decomposition, either B_t fully contains K_v or disjoint from it. Let us construct a tree decomposition $(T, (B'_t)_{t \in V(T)})$ of H_k such that B'_t contains v if and only if B_t fully contains K_v. It is easy to see that this is a tree decomposition of H_k: for every $E \in E(H_k)$, the set $\bigcup_{v \in E} K_v$ is a clique in G_k, hence there is a bag B_t containing $\bigcup_{v \in E} K_v$, i.e, B'_t contains E. Furthermore, $\phi_k(B'_t) < k$ for every bag B'_t: if $\phi_k(B'_t) \geq k$, then $|\bigcup_{v \in E} K_v| \geq q_k k$, contradicting the assumption that every bag B_t has size strictly less than $q_k k$. This would contradict the assumption $\mathrm{adw}(H_k) \geq k$, thus $\mathrm{tw}(G_k) \geq q_k k - 1$.

Simulating G_k by H_k. We present an algorithm for $\mathrm{CSP}(\mathcal{G})$ violating Conjecture 3. We show how a binary $\mathrm{CSP}(\mathcal{G})$ instance I_1 with graph G_k can be reduced to a $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ instance I_2 with hypergraph $H_k \in \mathcal{H}$. Then I_2 can be solved with the assumed polynomial-time algorithm for $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$. Let $G \in \mathcal{G}$ be the graph of the CSP instance I_1. By enumerating the hypergraphs in $\mathcal{H}$, we can find the first value k such that $G = G_k$. We construct a $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ instance I_2 with hypergraph H_k where every variable $v \in V(H_k)$ simulates the variables in K_v.

The domain $\mathrm{Dom}(v)$ of v is $D^{|K_v|}$, i.e., $\mathrm{Dom}(v)$ is the set of $|K_v|$-tuples of D. For every $v \in V(H_k)$, there is a natural bijection between the elements of $\mathrm{Dom}(v)$ and the $|D|^{|K_v|}$ possible assignments $f : K_v \to D$. For each edge $E = (v_1, \ldots, v_r) \in E(H_k)$, we add a constraint $c_E = \langle (v_1, \ldots, v_r), R_E \rangle$ to I_2 as follows. Consider a tuple $(x_1, \ldots, x_r) \in \prod_{i=1}^{r} \mathrm{Dom}(v_i)$. For $1 \leq i \leq r$, let g_i be the assignment of K_{v_i} corresponding to $x_i \in \mathrm{Dom}(v_i)$. These r assignments together define an assignment $g : \bigcup_{i=1}^{r} K_{v_i} \to D$ on the union of their domains. We define the relation R_E such that $(x_1, \ldots, x_r)$ is a member of R_E if and only if the corresponding assignment g satisfies every

constraint of I_1 whose scope is contained in $\bigcup_{i=1}^{r} K_{v_i}$.

Assume that I_1 has a solution $f_1 : V(G_k) \to D$. For every $v \in V(H_k)$, define $f_2(v)$ to be the member of $\mathrm{Dom}(v)$ corresponding to the assignment f_1 restricted to K_v. It is easy to see that f_2 is a solution of I_2: this follows from the fact that for every edge E of H_k, assignment f_1 restricted to $\bigcup_{v \in E} K_v$ clearly satisfies every constraint of I_1 whose scope is in $\bigcup_{v \in E} K_v$.

Assume now that I_2 has a solution $f_2 : V_2 \to D_2$. For every $v \in V(H_k)$, there is an assignment $f_v : K_v \to D$ corresponding to the value $f_2(v)$. These assignments together define an assignment $f_1 : V(G_k) \to D$. We claim that f_1 is a solution of I_1. Let $c = \langle (u', v'), R \rangle$ be an arbitrary constraint of I_1. Assume that $u' \in K_u$ and $v' \in K_v$ for some $u, v \in V(H_k)$. Since $u'v' \in E(G_k)$, there is an edge $E \in E(H_k)$ with $u, v \in E$. The definition of c_E in I_2 ensures that f_1 restricted to $K_u \cup K_v$ satisfies every constraint of I_1 whose scope is contained in $K_u \cup K_v$; in particular, f_1 satisfies constraint c.

Running time. Assume that an instance I_1 of $\mathrm{CSP}(\mathcal{G})$ is solved by first reducing it to an instance I_2 as above and then applying the algorithm for $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$. Let us determine the running time of this algorithm. The first step of the algorithm is to enumerate the hypergraphs in $\mathcal{H}$ until the correct value of k is found. The time required by this step depends only on the graph $G \in \mathcal{G}$; denote it by $h_2(G)$. Let us determine the time required to construct instance I_2 and the size of the representation of I_2. As defined above, for each constraint c_E in I_2, we have to enumerate every tuple $(x_1, \ldots, x_r) \in \prod_{i=1}^{r} \mathrm{Dom}(v)$ and check whether the corresponding assignment g satisfies every constraint whose scope is in $\bigcup_{i=1}^{r} K_{v_i}$. Checking a vector $(x_1, \ldots, x_r)$ can be done in time polynomial in $\|I_1\|$. Moreover,

$$\left| \prod_{i=1}^{r} \mathrm{Dom}(v) \right| = \prod_{i=1}^{r} |\mathrm{Dom}(v)| = \prod_{i=1}^{r} |D|^{|K_v|} = \prod_{i=1}^{r} |D|^{q_k \cdot \phi_k(v)} = |D|^{q_k \sum_{i=1}^{r} \phi_k(v)} \leq |D|^{q_k},$$

where the last inequality follows from the facts that ϕ_k is a fractional independent set and $\{x_1, \ldots, x_r\}$ is an edge of H_k. Every other step of the reduction can be done in time polynomial in $\|I_1\|$, hence the reduction can be done in time $h_2(G)\|I_1\|^{O(q_k)}$, which is also a bound on $\|I_2\|$. Thus the algorithm for $\mathrm{CSP}_{\mathrm{tt}}(\mathcal{H})$ requires $h_1(H_k)(h_2(G)\|I_1\|)^{O(q_k c)}$ time, yielding a total time of $h_3(G)\|I_1\|^{O(q_k c)}$ for some computable function h_3.

We show that $\|I_1\|^{O(q_k c)}$ is $\|I_1\|^{o(\mathrm{tw}(G_k))}$, violating Conjecture 3. Let $s(w)$ be the smallest k such that $\mathrm{tw}(G_k)$ is greater than w (as $\mathrm{tw}(G_k) \geq q_k k - 1$, the function $\mathrm{tw}(G_k)$ is unbounded, hence $s(w)$ is well defined). Observe that $s(w)$ is nondecreasing and unbounded. We have seen that $\mathrm{tw}(G) \geq q_k k - 1$. Thus

$$\|I_1\|^{O(q_k c)} \leq \|I_1\|^{O(c(\mathrm{tw}(G_k)+1)/k)} \leq \|I_1\|^{O(c(\mathrm{tw}(G)+1)/s(\mathrm{tw}(G)))} = \|I_1\|^{o(\mathrm{tw}(G))},$$

where the second inequality follows from the definition of s. Thus the total running time is $h_3(G)\|I_1\|^{o(\mathrm{tw}(G))}$, violating Conjecture 3. □

5 Separation of bounded fractional hypertree width and bounded adaptive width

We show that the class of sets of hypergraphs with bounded adaptive width strictly includes the class of sets with bounded fractional hypertree width. First, fractional hypertree width is an upper bound for adaptive width.

Proposition 15. *For every hypergraph H,* $\mathrm{adw}(H) \leq \mathrm{fhw}(H)$.

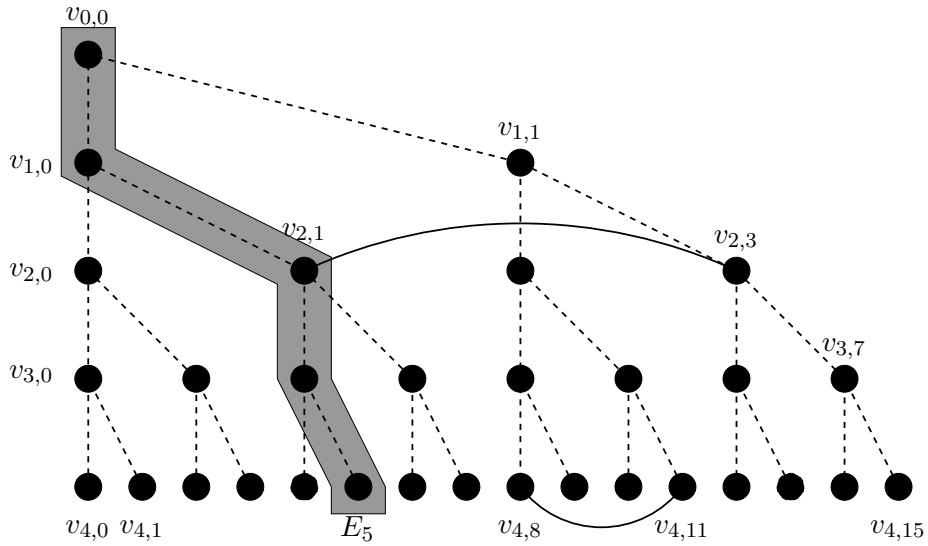

Figure 1: The structure of the hypergraph $H(4,3)$, showing the large edge E_5 and two small edges. Every large edge is a root to leaf path in the binary tree shown by dashed edges.

Proof. Let $(T, B_{t\in V(T)})$ be a tree decomposition of H whose ρ^*_H-width is fhw(H). If ϕ is a fractional independent set, then $\phi(B_t) \leq \rho^*_H(B_t) \leq$ fhw(H) for every bag B_t of the decomposition, i.e., ϕ-width$(H) \leq$ fhw(H). This is true for every fractional independent set ϕ, hence adw$(H) \leq$ fhw(H). □

This implies that if a set of hypergraphs has bounded fractional hypertree width, then it has bounded adaptive width as well. The converse is not true: the main result of this section is a set of hypergraphs with bounded adaptive width (Corollary 25) that has unbounded fractional hypertree width (Corollary 22).

Definition 16. The hypergraph $H(d,c)$ has $2^{d+1}-1$ vertices $v_{i,j}$ $(0 \leq i \leq d,\ 0 \leq j < 2^i)$ and the following edges:

- For every $0 \leq k < 2^d$, there is a *large edge* E_k of size $d+1$ that contains $v_{i,\lfloor k/2^{d-i}\rfloor}$ for every $0 \leq i \leq d$.
- For every i, j_1, j_2 with $|j_1 - j_2| \leq c$, there is a *small edge* $\{v_{i,j_1}, v_{i,j_2}\}$.

We say that vertex $v_{i,j}$ is on *level* i. Let us define $\chi(v_{i,j}) = j2^{d-i}$. The set $\mathcal{H}_c$ contains every hypergraph $H(d,c)$ for $d \geq 1$.

We can imagine the vertices of $H(d,c)$ as nodes of a complete binary tree on $d+1$ levels. For each leaf $v_{d,j}$ of the tree, the large edge $E_{\chi(v_{d,j})}$ contains the path from the leaf to the root $v_{0,0}$ (see Figure 1). The small edges connect some vertices on the same level. The χ-value of a vertex is the horizontal coordinate of the node in the figure.

The precise value of the parameter c is not very important to achieve the main result of the section: everything will work if we replace c with the constant 5.

Definition 17. If $v_{i,j}$ and $v_{i',j'}$ are covered by the same large edge E_k and $i \leq i'$, then $v_{i,j}$ is an *ancestor* of $v_{i',j'}$; and $v_{i',j'}$ is a *descendant* of $v_{i,j}$. We denote by $\mathcal{A}(v_{i,j})$ the set of ancestors of $v_{i,j}$ (observe that $v_{i,j} \in \mathcal{A}(v_{i,j})$ and $|\mathcal{A}(v_{i,j})| = i+1$).

Proposition 18. *If $v_{i,j}$ is an ancestor of $v_{i',j'}$, then $\chi(v_{i,j}) \leq \chi(v_{i',j'}) < \chi(v_{i,j}) + 2^{d-i}$.*

Proof. The unique ancestor of $v_{i',j'}$ on level i is $v_{i,\lfloor j'/2^{i'-i} \rfloor}$. Therefore,

$$\chi(v_{i,j}) = \lfloor j'/2^{i'-i} \rfloor \cdot 2^{d-i} \leq j' \cdot 2^{d-i'} = \chi(v_{i',j'})$$

and

$$\chi(v_{i,j}) = \lfloor j'/2^{i'-i} \rfloor \cdot 2^{d-i} > (j'/2^{i'-i} - 1) \cdot 2^{d-i} = j' \cdot 2^{d-i'} - 2^{d-i} = \chi(v_{i',j'}) - 2^{d-i}.$$

□

5.1 Lower bound on fractional hypertree width

Fractional hypertree width has various other characterizations that are equivalent up to a constant factor [14]. Here we use the characterization by balanced separators to prove a lower bound on the fractional hypertree width of $H(d, c)$.

For a function $\gamma : E(H) \to \mathbb{R}^+$, we define $\text{weight}(\gamma) := \sum_{E \in E(H)} \gamma(E)$. For a set $W \subseteq V(H)$, we let $\text{weight}(\gamma|W) = \sum_{e \in E_W} \gamma(e)$, where E_W is the set of all edges intersecting W. For $\lambda > 0$, a set $S \subseteq V(H)$ is a *λ-balanced separator for γ* if $\text{weight}(\gamma|C) \leq \lambda \cdot \text{weight}(\gamma)$ for every component C of $H \setminus S$.

Theorem 19 ([14]). *Let H be a hypergraph and $\gamma : E(H) \to \mathbb{R}^+$. There is a $\frac{1}{2}$-balanced separator S for γ such that $\rho^*_H(S) \leq \text{fhw}(H)$.*

Theorem 19 can be generalized to obtain λ-balanced separators with arbitrary $\lambda > 0$:

Corollary 20. *Let H be a hypergraph and $\gamma : E(H) \to \mathbb{R}^+$. For every $\lambda > 0$, there is a λ-balanced separator S for γ such that $\rho^*_H(S) \leq 2\,\text{fhw}(H)/\lambda$.*

Proof. For $\lambda \geq 1$, there is nothing to prove. We show that if the statement is true for 2λ, then it is also true for λ; this implies the validity of the statement for every $\lambda > 0$.

Let H be a hypergraph and $\gamma : E(H) \to \mathbb{R}^+$. If the statement is true for 2λ, then there is a 2λ-balanced separator S with fractional cover number at most $2\,\text{fhw}(H)/(2\lambda)$. Let $C_1, \ldots, C_t$ be the components of $H \setminus S$ with $\text{weight}(\gamma|C_i) > \lambda \cdot \text{weight}(\gamma)$; the weights imply that $t < 1/\lambda$. By Theorem 19, for each i, there is a $\frac{1}{2}$-balanced separator S_i of $H[C_i]$ for γ restricted to C_i. As $\text{weight}(\gamma|C_i) \leq 2\lambda \cdot \text{weight}(\gamma)$, every component C of $H[C_i \setminus S_i]$ has $\text{weight}(\gamma|C) \leq \lambda \cdot \text{weight}(\gamma)$. Thus $S \cup S_1 \cup \cdots \cup S_t$ is a λ-balanced separator of H for γ with fractional cover number at most

$$\text{fhw}(H)/\lambda + t\,\text{fhw}(H) < \text{fhw}(H)/\lambda + \text{fhw}(H)/\lambda = 2\,\text{fhw}(H)/\lambda,$$

as required. □

We prove the lower bound on the fractional hypertree width of $H(d, c)$ by presenting a function γ having no suitable λ-separator.

Proposition 21. *For every $c \geq 5$ and $d > 2\log_2 c$, we have $\text{fhw}(H(d, c)) \geq \sqrt{d}/(8c)$.*

Proof. Let γ be a weight function on the edges that assigns 1 to each large edge and 0 to the small edges. We show that every $\frac{1}{2c}$-balanced separator of $H(d, c)$ for γ has fractional cover number at least $\sqrt{d}/2$. By Corollary 20, this means that the fractional hypertree width is at least $\sqrt{d}/(8c)$.

Suppose that S is a $\frac{1}{2c}$-balanced separator of $H(d, c)$ for γ. Observe that on level $\lceil d/2 \rceil$, there are at least c vertices: $2^{\lceil d/2 \rceil} \geq c$. We claim that there is a $\lceil d/2 \rceil \leq i \leq d$ for which there is

no $0 \leq a_i \leq 2^i - c$ such that $v_{i,j} \in S$ for every $a_i \leq j < a_i + c$. Suppose that there is such an a_i for every $\lceil d/2 \rceil \leq i \leq d$. Let $b_i = a_i + c - 1$. It follows from the definition of a_i that $v_{i,a_i}, v_{i,b_i} \in S$ for every $\lceil d/2 \rceil \leq i \leq d$. We claim that the set $X = \{v_{i,a_i}, v_{i,b_i} : \lceil d/2 \rceil \leq i \leq d\}$ contains an independent set of size at least $\sqrt{d}/2$, proving that the fractional cover number $\rho^*(S)$ is at least $\sqrt{d}/2$ (recall that $\alpha_H(S) \leq \rho^*_H(S)$ holds). First we show that if a large edge E_k covers v_{i,a_i} and $v_{i',a_{i'}}$ then v_{i,b_i} and $v_{i',b_{i'}}$ are independent. Assume without loss of generality that $i < i'$. By Prop. 18, $|\chi(v_{i,a_i}) - \chi(v_{i',a_{i'}})| < 2^{d-i}$. Since $\chi(v_{i,b_i}) = \chi(v_{i,a_i}) + (c-1)2^{d-i}$ and $\chi(v_{i',b_{i'}}) = \chi(v_{i',a_{i'}}) + (c-1)2^{d-i'}$,

$$|\chi(v_{i,b_i}) - \chi(v_{i',b_{i'}})| \geq |\chi(v_{i,a_i}) - \chi(v_{i,b_i})| - |\chi(v_{i',a_{i'}}) - \chi(v_{i,a_i})| - |\chi(v_{i',b_{i'}}) - \chi(v_{i',a_{i'}})|$$
$$> (c-1)2^{d-i} - 2^{d-i} - (c-1)2^{d-i'} \geq (c-1)2^{d-i} - 2^{d-i} - \frac{c-1}{2} \cdot 2^{d-i} = (c/2 - 3/2)2^{d-i} \geq 2^{d-i},$$

if $c \geq 5$. Therefore, v_{i,b_i} and $v_{i',b_{i'}}$ are independent (Prop. 18). Similarly, if a large edge E_k covers both $v_{b_j,j}$ and $v_{b_{j'},j'}$ then $v_{a_j,j}$ and $v_{a_{j'},j'}$ are independent.

The size of X is $2(\lfloor d/2 \rfloor + 1) \geq d$. Therefore, if X can be covered with weight less than $\sqrt{d}/2$, then there has to be an edge that covers at least $|X|/(\sqrt{d}/2) = 2\sqrt{d}$ vertices of X. Denote by $Y \subseteq X$ this set of vertices, and let $Y_a = \{v_{i,a_i} \in Y : \lceil d/2 \rceil \leq i \leq d\}$ and $Y_b = \{v_{i,b_i} \in Y : \lceil d/2 \rceil \leq i \leq d\}$. Now either $|Y_a| \geq \sqrt{d}$ or $|Y_b| \geq \sqrt{d}$. For each vertex v_{i,a_i}, we call the vertex v_{i,b_i} the *partner* of v_{i,a_i} and vice versa. If $|Y_a| \geq \sqrt{d}$, then we have seen that the partners of the vertices in Y_a form an independent set of size $|Y_a|$, thus X cannot be covered with weight less than $\sqrt{d}$. Similarly, if $|Y_b| \geq \sqrt{d}$, then the partners of the vertices in Y_b give an independent set of size $\sqrt{d}$. This contradicts the assumption that S can be covered with weight strictly less than $\sqrt{d}/2$.

Thus there is a $d/2 \leq i \leq d$ such that for every $0 \leq j \leq 2^i - c$, at least one of $v_{i,j}$, ..., $v_{i,j+c-1}$ is not in S. Therefore, if C_i is the set of vertices on level i not in S, then C_i is a connected set (there are at least c vertices on level $i \geq d/2$, hence C_i is nonempty). This means that C_i contains more than $1/(2c)$ fraction of the vertices on level i, i.e., more than $2^i/(2c)$ vertices. Each vertex on level i is contained in 2^{d-i} large edges, hence the vertices in C_i are covered by more than $2^d/(2c)$ large edges. For the component C of $H(d,c)$ containing the connected set C, we have $\text{weight}(\gamma|C) > \text{weight}(\gamma)/2c$, contradicting the assumption that S is a $\frac{1}{2c}$-balanced separator for γ. □

Corollary 22. *$\mathcal{H}_c$ has unbounded fractional hypertree width for every $c \geq 5$.*

5.2 Upper bound on adaptive width

To show that $H(d,c)$ has small adaptive width, we have to show that it has small ϕ-width for every fractional independent set ϕ. The following lemma gives an upper bound on the f-width if certain balanced separators exist.

Lemma 23. *Let H be a hypergraph, $0 < \lambda < 1, w > 0$ constants, and $f : 2^{V(H)} \to \mathbb{R}^+$ a function such that*

- *$f(X) \leq f(Y)$ for every $X \subseteq Y$ (i.e., f is monotone) and*
- *$f(X \cup Y) \leq f(X) + f(Y)$ for arbitrary X, Y (i.e., f is subadditive).*

Assume that for every subset $W \subseteq V(H)$ there is a subset $S \subseteq V(H)$ with $f(S) \leq w$ such that every component C of $H \setminus S$ has $f(C \cap W) \leq \lambda f(W)$. Then the f-width of H is at most $2w/(1-\lambda) + w$.

Proof. We prove that if H and f satisfy the requirements, then H has a tree decomposition of f-width at most $2w(1-\lambda)+w$. More precisely, we prove the following stronger statement:

> For every subset $X \subseteq V(H)$ with $f(X) \leq 2w/(1-\lambda)$, the hypergraph H has a tree decomposition $\mathcal{T}$ of f-width at most $2w/(1-\lambda)+w$ where $X \subseteq B_t$ for some bag B_t of $\mathcal{T}$.

The proof is by induction on $|V(H)|$. Assume that the statement is true for every hypergraph with fewer vertices than H. If $f(V(H)) \leq 2w/(1-\lambda)+w$, then we are done, as a tree composition consisting of a single bag $B = V(H)$ satisfies the requirements. The requirements on H and f imply that $f(v) \leq w$ for every $v \in V(H)$ (consider the set $W := \{v\}$). Therefore, by adding new vertices to X one by one, we can obtain a superset $X' \supseteq X$ with $2w/(1-\lambda) < f(X') \leq 2w/(1-\lambda)+w$ (here we are using both monotonicity and subadditivity). By assumption, there is a set $S \subseteq V(H)$ with $f(S) \leq w$ such that if $C_1, \ldots, C_d$ are the components of $V(H) \setminus S$, then $f(C_i \cap X') \leq \lambda f(X') \leq \lambda w(2/(1-\lambda)+1)$ for every $1 \leq i \leq d$. Note that $d = 0$ is not possible, since that would imply $f(V(H)) \leq w$, which we have already excluded. For every $1 \leq i \leq d$, we have

$$\begin{aligned} f(C_i \cap X') + f(S) &\leq \lambda w(2/(1-\lambda)+1) + w \\ &= w(2\lambda + \lambda(1-\lambda) + (1-\lambda))/(1-\lambda) = w(2-(1-\lambda)^2)/(1-\lambda) < 2w/(1-\lambda). \end{aligned}$$

If $d = 1$, then $f(X') \leq f(C_1 \cap X') + f(S) < 2w/(1-\lambda)$ is a contradiction. Thus $d > 1$, and hence $H[C_i \cup S]$ has strictly fewer vertices than H for every $1 \leq i \leq d$. Set $X_i := (C_i \cap X) \cup S$ and observe that $f(X_i) \leq f(C_i \cap X) + f(S) \leq f(C_i \cap X') + f(S) < 2w/(1-\lambda)$ (using that f is monotone and subadditive). By the induction hypothesis, for every $1 \leq i \leq d$, the hypergraph $H[C_i \cup S]$ has a tree decomposition $\mathcal{T}_i$ of f-width at most $2w/(1-\lambda)+w$ such that some bag B_i of $\mathcal{T}_i$ fully contains X_i. We connect these d tree decompositions by introducing a new bag $B_0 := X \cup S$ that is the neighbor of bag B_i of $\mathcal{T}_i$ for every $1 \leq i \leq d$. It is easy to check that tree decomposition $\mathcal{T}$ obtained this way is a proper tree decomposition of H with f-width at most $2w/(1-\lambda)+w$ (note that $f(B_0) \leq f(X) + f(S) \leq 2w(1-\lambda)+w$). □

To obtain the upper bound on adaptive width using Lemma 23, we have to show that the required separator S exists for every fractional independent set. We say that a set S is *closed* if the set S contains every ancestor of every vertex of S, i.e., $\mathcal{A}(S) \subseteq S$. For future use, we show that even a closed separator exists for $H(d,c)$. This will imply that the proof of Lemma 23 can actually give a tree decomposition with the additional property that each bag is closed (Corollary 26).

Lemma 24. *Let ϕ be a fractional independent set of $H(d,c)$ and let W be a subset of vertices. Then there is a closed set S with $\phi(S) \leq 4c^2+8c+6$ such that for every component C of $H(d,c) \setminus S$ we have $\phi(C \cap W) \leq 3\phi(W)/4$.*

Proof. Let $M(a,b)$ be the set of vertices $v_{i,j}$ with $a \leq \chi(v_{i,j}) < b$. Our strategy is to find appropriate values t_1, t_2 such that separating $M(t_1,t_2)$ from the rest of the hypergraph gives a a balanced separator. We need to show that $M(t_1,t_2)$ can be separated by a set S such that $\phi(S)$ is small and $\phi(M(t_1,t_2) \cap W)$ is between $\phi(W)/4$ and $3\phi(W)/4$. The values t_1 and t_2 are found by an averaging argument: we argue that there have to be values where it is cheap to cut the hypergraph. An additional complication is that we have to treat the first few levels in a different way: roughly speaking, we can apply the averaging argument only if the neighborhood reachable by the small edges is small compared to the set of vertices we want to separate.

Let x and y be integers such that $\phi(M(x,x+y) \cap W) \geq \phi(W)/4$ and y is as small as possible. Let $d_0 = d - \lceil \log_2 y \rceil$; clearly, we have $y \leq 2^{d-d_0} \leq 2y$. Let $A(t) := \{v_{i,j} : \chi(v_{i,j}) \geq t \text{ and } i \geq d_0\}$.

Denote by $S(t)$ the set of those vertices v that have a descendant $v_{i,j}$ with $\chi(v_{i,j}) < t$ such that $v_{i,j}$ has a neighbor in $A(t)$. (Note that having a descendant $v_{i,j}$ with $\chi(v_{i,j}) < t$ immediately implies that $\chi(v) < t$ as well.) We show that $\phi(S(t_1)) \leq 2c^2 + 4c + 1$ for some $x - y < t_1 \leq x$.

Let $S_1(t)$ be those vertices of $S(t)$ that are on level less than d_0 and let $S_2(t)$ be those vertices that are on level at least d_0. First we bound $\phi(S_1(t))$. Let X be the set of vertices $v_{d_0,j}$ with $\chi(v_{d_0,j}) \in (t-(c+1)2^{d-d_0}, t)$. As the difference of the χ-values of any two vertices in X is at least 2^{d-d_0}, there can be at most $c+1$ such values in an open interval of length $(c+1)2^{d-d_0}$ and hence $|X| \leq c+1$ follows. We show that every vertex of $S_1(t)$ has a descendant in X. This means that if we cover the $c+1$ vertices in X with $c+1$ large edges, then every vertex of $S_1(t)$ is covered. As $\phi(E_k) \leq 1$ for every large edge (since ϕ is a fractional independent set), $\phi(S_1(t)) \leq c+1$ follows.

To show that every vertex $v_{i',j'} \in S_1(t)$ has a descendant in X, we need to consider two cases. Suppose first that $v_{i',j'}$ itself has a descendant in $u \in A(t)$. By Prop 18, the ancestor of u on level d_0 has χ-value greater than $t - 2^{d-d_0}$, thus $v_{i',j'}$ has a descendant on level d_0 with χ-value greater than $t - 2^{d-d_0}$. Furthermore, $\chi(v_{i',j'}) < t$ implies that $v_{i',j'}$ has a descendant on level d_0 with χ-value less than t. Thus if u' is the descendant of $v_{i',j'}$ on level d_0 with $\chi(u') < t$ having maximum χ-value, then $\chi(u')$ is in the interval $[t - 2^{d-d_0}, t)$, which means that $u' \in X$.

If $v_{i',j'}$ has no descendant in $A(t)$, then it has a descendant $v_{i,j}$ with $\chi(v_{i,j}) < t$ that is connected to a vertex of $A(t)$ with a small edge. This means that $\chi(v_{i,j}) \geq t - c2^{d-i}$. If u is the ancestor of $v_{i,j}$ on level d_0, then by Prop. 18,

$$\chi(u) > \chi(v_{i',j'}) - 2^{d-d_0} \geq t - c(2^{d-i}) - 2^{d-d_0} \geq t - (c+1)2^{d-d_0},$$

implying $v_{d_0,j''} \in X$. Thus we showed that every vertex of $S_1(t)$ has a descendant in X and $\phi(S_1(t)) \leq c+1$ is proved.

We cannot bound $\phi(S_2(t))$ uniformly for every value of t, but we show that it is small on average. We claim that

$$\sum_{t=x-y+1}^{x} \phi(S_2(t)) \leq c \sum_{t=\max\{0,(x-y-c2^{d-d_0})\}}^{\min\{2^d-1, x+2^{d-d_0}-1\}} \phi(E_t) \leq c\Big((c+1)2^{d-d_0} + y\Big) \leq 2c(c+1)y + cy.$$

holds, implying that $\phi(S_2(t)) \leq 2c(c+1) + c = 2c^2 + 3c$ and hence $\phi(S(t)) \leq 2c^2 + 4c + 1$ for at least one value of t. To see the first inequality, observe that $v_{i,j}$ with $i \geq d_0$ is in $S_2(t)$ only if $t - c2^{d-i} \leq \chi(v_{i,j}) < t$. Thus such a vertex contributes to the first sum for at most $c2^{d-i}$ values of t. However, if $v_{i,j}$ contributes at all to the first sum (which is only possible if $x - y - c2^{d-i} \leq \chi(v_{i,j}) < x$), then it contributes to the second sum for exactly 2^{d-i} values of t, as every large edge containing $v_{i,j}$ is counted. Thus every vertex $v_{i,j}$ contributes $\phi(v_{i,j})$ at most c times more to the first sum than to the second, which is taken care by the factor c before the second sum. The second inequality follows from the fact that ϕ is a fractional independent set, i.e, $\phi(E_t) \leq 1$. The last inequality follows from the definition of d_0.

Similarly, we can show that there is a value $x+y \leq t_2 < x+2y$ such that $\phi(S_2(t_2)) \leq 2c^2+4c+1$. Denote by $T(t_1, t_2)$ the vertices of $M(t_1, t_2)$ on levels less than d_0. We claim that $\phi(T(t_1, t_2)) \leq 3$ (if $d_0 = 0$, then there is nothing to show). First, $T(t_1, t_2)$ can contain at most 3 vertices on each level: if $v_{i,j}, v_{i,j'} \in T(t_1, t_2)$ and $j' \geq j+3$, then $|\chi(v_{i,j}) - \chi(v_{i,j'})| \geq 3 \cdot 2^{d-i} > 3 \cdot 2^{d-d_0} \geq 3y \geq t_2 - t_1$, contradicting the assumption on the χ-values. Every $v_{i,j} \in T(t_1, t_2)$ has a descendant $v_{i',j'} \in T(t_1, t_2)$ for every $i \leq i' < d_0$, namely $v_{i',j'}$ with $j' = j2^{i'-i}$. In particular, this means that every vertex of $T(t_1, t_2)$ is an ancestor of one of the at most 3 vertices of $T(t_1, t_2)$ on level $d_0 - 1$. Therefore, if we cover these vertices on level $d_0 - 1$ with at most 3 large edges, then the whole set $T(t_1, t_2)$ is covered. As $\phi(E_t) \leq 1$ for every large edge E_t, it follows that $\phi(T(t_1, t_2)) \leq 3$.

We obtain the set S required by lemma by setting

$$S := S(t_1) \cup S(t_2) \cup T(t_1, t_2) \cup E_{x+y}.$$

Clearly, $\phi(S) \leq 2(2c^2 + 4c + 1) + 3 + 1 = 4c^2 + 8c + 6$, as required by the lemma. We show that S separates $M(t_1, t_2)$ from the rest of the vertices. Suppose that $v_{i,j}, v_{i',j'} \notin S$ are adjacent vertices such that $v_{i,j} \in M(t_1, t_2)$ and $v_{i',j'} \notin M(t_1, t_2)$. We have $i \geq d_0$ (otherwise $v_{i,j} \in T(t_1, t_2) \subseteq S$), hence $v_{i,j} \in A(t_1)$. If $\chi(v_{i',j'}) < t_1$, then $v_{i',j'} \in S(t_1) \subseteq S$, a contradiction. Moreover, if $\chi(v_{i',j'}) > t_2$, then $i' \geq d_0$ as $v_{i,j}$ and $v_{i',j'}$ are not neighbors if $\chi(v_{i,j}) < \chi(v_{i',j'})$ and $i > i'$. Thus $v_{i',j'} \in A(t_2)$ and $v_{i,j} \in S(t_2) \subseteq S$, a contradiction.

By the definition of x and y, we have $\phi(M(t_1, t_2) \cap W) \geq \phi(M(x, x + y) \cap W) \geq \phi(W)/4$. To complete the proof that $\phi(W \cap C) \leq 3\phi(W)/4$ for every component C of $H(d, c) \setminus S$, we need to show that $\phi((M(t_1, t_2) \setminus S) \cap W) \leq 3\phi(W)/4$: as we have seen that every such component C is either fully contained in $M(t_1, t_2)$ or disjoint from $M(t_1, t_2)$, this means that no component C can have $\phi(W \cap C) > 3\phi(W)/4$. Since $x - t_1 < y$, the minimality of y implies $\phi(M(t_1, x) \cap W) \leq \phi(W)/4$. Similarly, it follows from $t_2 - (x + y) < y$ that $\phi(M(x + y, t_2) \cap W) \leq \phi(W)/4$. The set $E_{x+y} \subseteq S$ fully covers every vertex v with $\chi(v) = x + y$ and (again by the minimality of y), we have $\phi(M(x, x + y - 1)) \leq \phi(W)/4$. Now

$$\phi((M(t_1, t_2) \setminus S) \cap W) \leq \phi(M(t_1, x) \cap W) + \phi(M(x, x+y-1) \cap W) + \phi(M(x+y, t_2) \cap W) \leq \frac{3}{4}\phi(W).$$

□

By Lemma 24, the requirements of Lemma 23 hold for $H(d, c)$ with $w := 4c^2 + 8c + 6$ and $\lambda := 3/4$, hence $\operatorname{adw}(H(d, c)) \leq 9w = 36c^2 + 72c + 54$, i.e., it can be bounded by a constant depending only on c, but not on d.

Corollary 25. *The class $\mathcal{H}_c$ has bounded adaptive width for every fixed $c \geq 1$.*

For future use, we argue that the width $36c^2 + 72c + 54$ can be reached with a tree decomposition where each bag is closed.

Corollary 26. *For every fractional independent set ϕ of $H(d, c)$, the hypergraph $H(d, c)$ has a tree decomposition with ϕ-width at most $36c^2 + 54c + 72$ such that every bag is closed.*

Proof. Let us go through the proof of Lemma 23 and show what further arguments are needed to claim that every bag is closed in the resulting tree decomposition. We change the induction statement to

> For every *closed* subset $X \subseteq V(H)$ with $f(X) \leq 2w/(1 - \lambda)$, hypergraph H has a tree decomposition $\mathcal{T}$ of f-width at most $2w/(1 - \lambda) + w$ where $X \subseteq B_t$ for some bag B_t of $\mathcal{T}$ *and every bag of $\mathcal{T}$ is closed.*

We can assume that S is closed. We claim that $X_i := (C_i \cap X) \cup S$ is closed. Since X is closed, we have $\mathcal{A}(C_i \cap X) \subseteq X$. Every vertex of $\mathcal{A}(C_i \cap X) \setminus (C_i \cap X)$ is adjacent to $C_i \cap X$, thus it is either in S or in $C_i \cap X$. It follows that $\mathcal{A}(C_i \cap X) \subseteq (C_i \cap X) \cup S$ and $\mathcal{A}(X_i) \subseteq X_i$. This means that X_i is closed and hence the induction hypothesis can be applied on X_i. Finally, $B_0 := X \cup S$ is closed (the union of two closed set is closed), and by the induction statement, every other bag is closed as well. □

References

[1] I. Adler. *Width functions for hypertree decompositions.* PhD thesis, Albert-Ludwigs-Universität Freiburg, 2006.

[2] A. A. Bulatov. Tractable conservative constraint satisfaction problems. In *18th Annual IEEE Symposium on Logic in Computer Science (LICS'03)*, page 321, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[3] A. A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a 3-element set. *J. ACM*, 53(1):66–120, 2006.

[4] A. A. Bulatov, A. A. Krokhin, and P. Jeavons. The complexity of maximal constraint languages. In *Proceedings of the 33rd ACM Symposium on Theory of Computing*, pages 667–674, 2001.

[5] H. Chen and M. Grohe. Constraint satisfaction problems with succinctly specified relations, 2006. Manuscript. Preliminary version in *Dagstuhl Seminar Proceedings 06401: Complexity of Constraints.*

[6] R. G. Downey and M. R. Fellows. *Parameterized Complexity.* Monographs in Computer Science. Springer, New York, 1999.

[7] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM J. Comput.*, 28(1):57–104, 1999.

[8] J. Flum and M. Grohe. *Parameterized Complexity Theory.* Springer, Berlin, 2006.

[9] E. C. Freuder. Complexity of k-tree structured constraint satisfaction problems. In *Proc. of AAAI-90*, pages 4–9, Boston, MA, 1990.

[10] G. Gottlob, N. Leone, and F. Scarcello. Hypertree decompositions and tractable queries. *Journal of Computer and System Sciences*, 64:579–627, 2002.

[11] G. Gottlob, F. Scarcello, and M. Sideri. Fixed-parameter complexity in AI and nonmonotonic reasoning. *Artificial Intelligence*, 138(1-2):55–86, 2002.

[12] M. Grohe. The structure of tractable constraint satisfaction problems. In *MFCS 2006*, pages 58–72, 2006.

[13] M. Grohe. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM*, 54(1):1, 2007.

[14] M. Grohe and D. Marx. Constraint solving via fractional edge covers. In *SODA '06: Proceedings of the 17th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 289–298, New York, NY, USA, 2006. ACM Press.

[15] M. Grohe, T. Schwentick, and L. Segoufin. When is the evaluation of conjunctive queries tractable? In *STOC '01: Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 657–666, New York, NY, USA, 2001. ACM Press.

[16] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? *J. Comput. System Sci.*, 63(4):512–530, 2001.

[17] P. Jeavons, D. A. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

[18] P. G. Kolaitis and M. Y. Vardi. Conjunctive-query containment and constraint satisfaction. *J. Comput. Syst. Sci.*, 61(2):302–332, 2000.

[19] D. Marx. Can you beat treewidth? In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 169–179, 2007.

[20] D. Marx. Approximating fractional hypetree width. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'09)*, 2009.

[21] F. Scarcello, G. Gottlob, and G. Greco. Uniform constraint satisfaction problems and database theory. In *Complexity of Constraints*, pages 156–195, 2008.

[22] T. J. Schaefer. The complexity of satisfiability problems. In *Conference Record of the Tenth Annual ACM Symposium on Theory of Computing (San Diego, Calif., 1978)*, pages 216–226. ACM, New York, 1978.