Complexity Analysis of Generalized and Fractional Hypertree Decompositions

GEORG GOTTLOB and MATTHIAS LANZINGER, University of Oxford, United Kingdom and TU Wien, Austria
REINHARD PICHLER, TU Wien, Austria
IGOR RAZGON, Birkbeck University of London, United Kingdom

Hypertree decompositions (HDs), as well as the more powerful generalized hypertree decompositions (GHDs), and the yet more general fractional hypertree decompositions (FHDs) are hypergraph decomposition methods successfully used for answering conjunctive queries and for solving constraint satisfaction problems. Every hypergraph H has a width relative to each of these methods: its hypertree width $hw(H)$, its generalized hypertree width $ghw(H)$, and its fractional hypertree width $fhw(H)$, respectively. It is known that $hw(H) \leq k$ can be checked in polynomial time for fixed k, while checking $ghw(H) \leq k$ is NP-complete for $k \geq 3$. The complexity of checking $fhw(H) \leq k$ for a fixed k has been open for over a decade.

We settle this open problem by showing that checking $fhw(H) \leq k$ is NP-complete, even for $k = 2$. The same construction allows us to prove also the NP-completeness of checking $ghw(H) \leq k$ for $k = 2$. After that, we identify meaningful restrictions that make checking for bounded ghw or fhw tractable or allow for an efficient approximation of the fhw.

CCS Concepts: • **Information systems → Relational database query languages**; • **Theory of computation → Problems, reductions and completeness;**

Additional Key Words and Phrases: Generalized hypertree decompositions, fractional hypertree decompositions, np-hardness, tractable fragments, multi-intersection width, hypergraphs, candidate tree decompositions

ACM Reference format:
Georg Gottlob, Matthias Lanzinger, Reinhard Pichler, and Igor Razgon. 2021. Complexity Analysis of Generalized and Fractional Hypertree Decompositions. *J. ACM* 68, 5, Article 38 (September 2021), 50 pages.
https://doi.org/10.1145/3457374

This is a significantly extended and enhanced version of a paper presented at PODS 2018 [24].

The authors acknowledge support by the Austrian Science Fund (FWF): Projects P30930 and Y698. Georg Gottlob is a Royal Society Research Professor and acknowledges support by the Royal Society in this role through the "RAISON DATA" project (Reference No. RP\R1\201074). Matthias Lanzinger acknowledges support by the Royal Society "RAISON DATA" project (Reference No. RP\R1\201074).

Authors' addresses: G. Gottlob and M. Lanzinger, University of Oxford, Department of Computer Science, Oxford, UK, TU Wien, Vienna, Austria; emails: {georg.gottlob, matthias.lanzinger}@cs.ox.ac.uk; R. Pichler, TU Wien, Vienna, Austria; email: reinhard.pichler@tuwien.ac.at; I. Razgon, Birkbeck University of London, London, UK; email: igor@dcs.bbk.ac.uk.

0004-5411/2021/09-ART38 $15.00
https://doi.org/10.1145/3457374

1 INTRODUCTION AND BACKGROUND

Research Challenges Tackled. In this work, we tackle computational problems on hypergraph decompositions, which play a prominent role for answering **Conjunctive Queries (CQs)** and solving **Constraint Satisfaction Problems (CSPs)**, which we discuss below.

Many NP-hard graph-based problems become tractable for instances whose corresponding graphs have bounded treewidth. There are, however, many problems for which the structure of an instance is better described by a hypergraph than by a graph, for example, the above mentioned CQs and CSPs. Treewidth does not generalize hypergraph acyclicity (sometimes also referred to as α-acyclicity, see e.g., References [22, 54]). Hence, proper hypergraph decomposition methods have been developed, in particular, ***hypertree decompositions (HDs)*** [31], the more general ***generalized hypertree decompositions (GHDs)*** [31], and the yet more general ***fractional hypertree decompositions (FHDs)*** [35], and corresponding notions of width of a hypergraph H have been defined: the *hypertree width* $hw(H)$, the *generalized hypertree width* $ghw(H)$, and the *fractional hypertree width* $fhw(H)$, where for every hypergraph H, $fhw(H) \leq ghw(H) \leq hw(H)$ holds. Definitions are given in Section 2. A number of highly relevant hypergraph-based problems such as CQ-evaluation and CSP-solving become tractable for classes of instances of bounded hw, ghw, or, fhw. For each of the mentioned types of decompositions it would thus be useful to be able to recognize for each constant k whether a given hypergraph H has corresponding width at most k, and if so, to compute such a decomposition. More formally, for *decomposition* ∈ {HD, GHD, FHD} and $k > 0$, we consider the following family of problems:

Check(*decomposition*, k)
input: hypergraph $H = (V, E)$;
output: *decomposition* of H of width $\leq k$ if it exists and
answer “no” otherwise.

As shown in Reference [31], Check(HD, k) is in Ptime for every fixed k. However, little is known about Check(FHD, k). In fact, this has been an open problem since the 2006 paper [34], where Grohe and Marx state: “It remains an important open question whether there is a polynomial-time algorithm that determines (or approximates) the fractional hypertree width and constructs a corresponding decomposition.” Since then, the approximation problem has been resolved [44]. Regarding the problem of determining the exact width, the 2014 journal version [35] still mentions this as open and it is conjectured that the problem might be NP-hard. The open problem is restated in Reference [51], where further evidence for the hardness of the problem is given by showing that “it is not expressible in monadic second-order logic whether a hypergraph has bounded (fractional, generalized) hypertree width.” We will tackle this open problem here:

Research Challenge 1: Is Check(FHD, k) tractable?

Let us now turn to generalized hypertree decompositions. In Reference [31] the complexity of Check(GHD, k) was stated as an open problem. In Reference [32], it was shown that Check(GHD, k) is NP-complete for $k \geq 3$. For $k = 1$ the problem is trivially tractable because $ghw(H) = 1$ just means H is acyclic. However, the case $k = 2$ has been left open. This case is quite interesting, because it was observed that the majority of practical queries from various benchmarks that are not acyclic have $ghw = 2$ [11, 23], and that a decomposition in such cases can be very helpful. Our second research goal is to finally settle the complexity of Check(GHD, k) completely.

Research Challenge 2: Is Check(GHD, 2) tractable?

For those problems that are known to be intractable, for example, Check(GHD, k) for $k \geq 3$, and for those others that will turn out to be intractable, we would like to find large islands of tractability that correspond to meaningful restrictions of the input hypergraph instances. Ideally, such restrictions should fulfill two main criteria: (i) they need to be *realistic* in the sense that they apply to a large number of CQs and/or CSPs in real-life applications, and (ii) they need to be *non-trivial* in the sense that the restriction itself does not already imply bounded *hw*, *ghw*, or *fhw*. Trivial restrictions would be, for example, acyclicity or bounded treewidth. Hence, our third research problem is as follows:

> **Research Challenge 3:** Find realistic, non-trivial restrictions on hypergraphs that entail the tractability of the Check($decomp, k$) problem for $decomp \in \{\text{GHD}, \text{FHD}\}$.

Where we do not achieve Ptime algorithms for the precise computation of a decomposition of optimal width, we would like to find tractable methods for achieving good approximations. Note that for GHDs, the problem of approximations is solved, since $ghw(H) \leq 3 \cdot hw(H) + 1$ holds for every hypergraph H [4]. In contrast, for FHDs, the best known polynomial-time approximation is cubic. More precisely, in Reference [44], a polynomial-time algorithm is presented that, given a hypergraph H with $fhw(H) = k$, computes an FHD of width $O(k^3)$. We would like to find meaningful restrictions that guarantee significantly tighter approximations in polynomial time. This leads to the fourth research problem:

> **Research Challenge 4:** Find realistic, non-trivial restrictions on hypergraphs that allow us to compute in Ptime good approximations of $fhw(k)$.

Background and Applications. Hypergraph decompositions have meanwhile found their way into commercial database systems such as LogicBlox [6, 10, 40, 41, 48] and advanced research prototypes such as EmptyHeaded [1, 2, 50]. Moreover, since CQs and CSPs of bounded hypertree width fall into the highly parallelizable complexity class LogCFL [31], hypergraph decompositions have also been discovered as a useful tool for parallel query processing with MapReduce [5]. Hypergraph decompositions, in particular, HDs and GHDs, have been used in many other contexts, e.g., in combinatorial auctions [28] and automated selection of Web services based on recommendations from social networks [39]. Exact algorithms for computing the generalized and fractional hypertree width were published, for example, in Reference [47]; these algorithms require exponential time, which, in the light of Reference [32] and our present results cannot be improved in the general case.

CQs are the most basic and arguably the most important class of queries in the database world. Likewise, CSPs constitute one of the most fundamental classes of problems in Artificial Intelligence. Formally, CQs and CSPs are the same problem and correspond to first-order formulae using $\{\exists, \wedge\}$ but disallowing $\{\forall, \vee, \neg\}$ as connectives, which need to be evaluated over a set of finite relations: the *database relations* for CQs, and the *constraint relations* for CSPs. In practice, CQs have often fewer conjuncts (query atoms) and larger relations, while CSPs have more conjuncts but smaller relations. These problems are well-known to be NP-complete [13]. Consequently, there has been an intensive search for tractable fragments of CQs and/or CSPs over the past decades. For our work, the approaches based on decomposing the structure of a given CQ or CSP are most relevant, see, e.g., References [8, 14, 15, 17–19, 25, 31, 33, 35–38, 42, 45, 46]. The underlying structure of both is nicely captured by hypergraphs. The hypergraph $H = (V(H), E(H))$ underlying a CQ (or a CSP) Q has as vertex set $V(H)$ the set of variables occurring in Q; moreover, for every atom in Q, $E(H)$ contains a hyperedge consisting of all variables occurring in this atom. From now on, we shall mainly talk about hypergraphs with the understanding that all our results are equally applicable to CQs and CSPs.

Main Results. First, we have investigated the above-mentioned open problem concerning the recognizability of $fhw \leq k$ for fixed k. Our initial hope was to find a simple adaptation of the

NP-hardness proof in Reference [32] for recognizing $ghw(H) \leq k$, for $k \geq 3$. Unfortunately, this proof dramatically fails for the fractional case. In fact, the hypergraph-gadgets in that proof are such that both "yes" and "no" instances may yield the same *fhw*. However, via crucial modifications, including the introduction of novel gadgets, we succeed to construct a reduction from 3SAT that allows us to control the *fhw* of the resulting hypergraphs such that those hypergraphs arising from "yes" 3SAT instances have $fhw(H) = 2$ and those arising from "no" instances have $fhw(H) > 2$. Surprisingly, thanks to our new gadgets, the resulting proof is actually significantly simpler than the NP-hardness proof for recognizing $ghw(H) \leq k$ in Reference [32]. We thus obtain the following result:

> **Main Result 1:** Deciding $fhw(H) \leq 2$ for hypergraphs H is NP-complete and, therefore, Check(FHD, k) is intractable even for $k = 2$.

This result can be extended to the NP-hardness of recognizing $fhw(H) \leq k$ for arbitrarily large k. Moreover, the same construction can be used to prove that recognizing ghw ≤2 is also NP-hard, thus killing two birds with one stone.

> **Main Result 2:** Deciding $ghw(H) \leq 2$ for hypergraphs H is NP-complete and, therefore, Check(GHD, 2) is intractable even for $k = 2$.

The Main Results 1 and 2 are presented in Section 3. These results close some smoldering open problems with bad news. We thus further concentrate on Research Challenges 3 and 4 to obtain some positive results for restricted hypergraph classes.

We first study GHDs, where we succeed to identify very general, realistic, and non-trivial restrictions that make the Check(GHD, k) problem tractable. These results are based on new insights about the differences of GHDs and HDs and conceptually splitting the problem into two tasks: The first problem, for a given list of possible bags of vertices and a hypergraph H, consists in checking whether there exists a tree decomposition of H using only bags from the input. In Section 4, we show that this problem is NP-complete in general but becomes tractable when we introduce a mild restriction on the tree decompositions. The second problem consists in finding restrictions under which we only need to consider a polynomial number of possible bags.

In particular, we concentrate on the ***bounded intersection property (BIP)***, which, for a class $\mathscr{C}$ of hypergraphs requires that for some constant i, for each pair of distinct edges e_1 and e_2 of each hypergraph $H \in \mathscr{C}$, $|e_1 \cap e_2| \leq i$, and its generalization, the ***bounded multi-intersection property (BMIP)***, which requires that for some constant c any intersection of c distinct hyperedges of H has at most i elements for some constant i. A recent empirical study [23] of a large number of known CQ and CSP benchmarks showed that a high portion of instances coming from real-life applications indeed enjoys the BIP for low constant i and a yet higher portion enjoys the BMIP for very low constants c and i. We obtain the following favorable results, which are presented in Section 5:

> **Main Result 3:** For classes of hypergraphs fulfilling the BIP or BMIP, for every constant k, the problem Check(GHD, k) is tractable. Tractability holds even for classes $\mathscr{C}$ of hypergraphs where for some constant c all intersections of c distinct edges of every $H \in \mathscr{C}$ of size n have $O(\log n)$ elements. Our complexity analysis reveals that for fixed k and c the problem Check(GHD, k) is fixed-parameter tractable parameterized by i of the BMIP.

The tractability proofs for GHDs do not directly carry over to FHDs. Still, under slightly less general conditions and with some additional combinatorial insights it is possible to reduce the Check(FHD, k) problem to the Check(GHD, k) scenario of the previous result. In particular, we obtain results for the BIP and a further special case of the BMIP. We then consider the *degree d* of a hypergraph $H = (V(H), E(H))$, which is defined as the maximum number of hyperedges in which a vertex occurs, i.e., $d = \max_{v \in V(H)} |\{e \in E(H) \mid v \in E(H)\}|$. We say that a class $\mathscr{C}$ of hypergraphs has the ***bounded degree property (BDP)*** if there exists $d \geq 1$, such that every hypergraph $H \in \mathscr{C}$ has degree $\leq d$. We obtain the following results, which are presented in Section 6:

> **Main Result 4:** For classes of hypergraphs fulfilling either the BDP or the BIP and for every constant k, the problem Check(FHD, k) is tractable.

To get yet bigger tractable classes, we also consider approximations of an optimal FHD. Towards this goal, we study the *fhw* in case of the BMIP and we establish an interesting connection between the BMIP and the **Vapnik–Chervonenkis dimension (VC-dimension)** of hypergraphs. Our research, presented in Section 6 is summarized as follows:

> **Main Result 5:** For rather general, realistic, and non-trivial hypergraph restrictions, there exist Ptime algorithms that, for hypergraphs H with $fhw(H) = k$, where k is a constant, produce FHDs whose widths are significantly smaller than the best previously known approximation. In particular, the BMIP allows us to compute in polynomial time an FHD whose width is $\leq k+\epsilon$ for arbitrarily chosen constant $\epsilon > 0$. Bounded VC-dimension allows us to compute in polynomial time an FHD whose width is $O(k \log k)$.

We finally turn our attention also to the optimization problem of fractional hypertree width, i.e., given a hypergraph H, determine $fhw(H)$ and find an FHD of width $fhw(H)$. All our algorithms for the Check(FHD, k) problem have a runtime exponential in the desired width k. Hence, even with the restrictions to the BIP or BMIP, we cannot expect an efficient approximation of *fhw* if *fhw* can become arbitrarily large. We will therefore study the following K-Bounded-FHW-Optimization problem for constant $K \geq 1$:

K-Bounded-FHW-Optimization

input:	hypergraph $H = (V, E)$;	
output:	if $fhw(H) \leq K$:	find an FHD $\mathcal{F}$ of H with minimum width;
	otherwise:	answer "$fhw(H) > K$."

For this bounded version of the optimization problem, we will prove the following result:

> **Main Result 6:** There exists a polynomial time approximation scheme (PTAS; for details, see Section 6) for the K-Bounded-FHW-Optimization problem in case of the BMIP for any fixed $K \geq 1$.

2 PRELIMINARIES

For integers $n \geq 1$, we write $[n]$ to denote the set $\{1, \ldots, n\}$.

2.1 Hypergraphs

A *hypergraph* is a pair $H = (V(H), E(H))$, consisting of a set $V(H)$ of *vertices* and a set $E(H)$ of *hyperedges* (or, simply *edges*), which are non-empty subsets of $V(H)$. We assume that hypergraphs do not have isolated vertices, i.e., for each $v \in V(H)$, there is at least one edge $e \in E(H)$, s.t. $v \in e$. For a set $C \subseteq V(H)$, we define $edges(C) = \{e \in E(H) \mid e \cap C \neq \emptyset\}$ and for a set $S \subseteq E(H)$, we define $V(S) = \{v \in e \mid e \in S\}$. The *rank* of a hypergraph H (denoted $rank(H)$) is the maximum cardinality of any edge e of H. We refer to the number of edges and vertices as $|E(H)|$ and $|V(H)|$, respectively. The *size* of (some reasonable representation of) H will be denoted as $||H||$, i.e., $n \leq |V(H)| + |E(H)| \cdot |V(H)|$.

We sometimes identify sets of edges with hypergraphs. If a set of edges E is used, where instead a hypergraph is expected, then we mean the hypergraph (V, E), where V is simply the union of all edges in E. For a set S of edges, it is convenient to write $\bigcup S$ (and $\bigcap S$, respectively) to denote the set of vertices obtained by taking the union (or the intersection, respectively) of the edges in S. Hence, we can write $V(S)$ simply as $\bigcup S$.

For a hypergraph H and a set $C \subseteq V(H)$, we say that a $[C]$-path π from v to v' consists of a sequence $v = v_0, \dots, v_h = v'$ of vertices and a sequence of edges $e_0, \dots, e_{h-1}$ ($h \geq 0$) such that $\{v_i, v_{i+1}\} \subseteq (e_i \backslash C)$, for each $i \in \{0, \dots, h-1\}$. We denote by $V(\pi)$ the set of vertices occurring in the sequence $v_0, \dots, v_h$. Likewise, we denote by $edges(\pi)$ the set of edges occurring in the sequence $e_0, \dots, e_{h-1}$. A set $W \subseteq V(H)$ of vertices is $[C]$-connected if $\forall v, v' \in W$ there is a $[C]$-path from v to v'. A $[C]$-component is a maximal $[C]$-connected, non-empty set of vertices $W \subseteq V(H) \setminus C$.

The *primal graph* G of a hypergraph H is the graph with the same vertices as H and an edge between vertices v and u iff there exists an edge $e \in E(H)$ such that $\{v, u\} \subseteq e$.

Given a hypergraph $H = \{V, E\}$, the *dual hypergraph* $H^d = (W, F)$ is defined as $W = E$ and $F = \{\{e \in E \mid v \in e\} \mid v \in V\}$.

2.2 (Fractional) Edge Covers

Let $H = (V(H), E(H))$ be a hypergraph and consider (edge-weight) functions $\lambda \colon E(H) \rightarrow \{0, 1\}$ and $\gamma \colon E(H) \rightarrow [0, 1]$. For $\theta \in \{\lambda, \gamma\}$, we denote by $B(\theta)$ the set of all vertices *covered* by θ:

$$B(\theta) = \left\{ v \in V(H) \mid \sum_{e \in E(H),\, v \in e} \theta(e) \geq 1 \right\}.$$

The weight of such a function θ is defined as

$$weight(\theta) = \sum_{e \in E(H)} \theta(e).$$

Following Reference [31], we will sometimes consider λ as a set with $\lambda \subseteq E(H)$ (i.e., the set of edges e with $\lambda(e) = 1$) and the weight of λ as the cardinality of this set. However, for the sake of a uniform treatment with function γ, we shall prefer to treat λ as a function.

Definition 2.1. An *edge cover* of a hypergraph H is a function $\lambda : E(H) \rightarrow \{0, 1\}$ such that $V(H) = B(\lambda)$. The *edge cover number* $\rho(H)$ is the minimum weight of all edge covers of H.

Note that the edge cover number can be calculated by the following **integer linear program (ILP)**:

$$\begin{aligned} \text{minimize:} \quad & \sum_{e \in E(H)} \lambda(e) & \\ \text{subject to:} \quad & \sum_{e \in E(H),\, v \in e} \lambda(e) \geq 1, & \text{for all } v \in V(H) \\ & \lambda(e) \in \{0, 1\} & \text{for all } e \in E(H). \end{aligned}$$

By substituting all $\lambda(e)$ by $\gamma(e)$ and by relaxing the last condition of the ILP above to $\gamma(e) \geq 0$, we arrive at the **linear program (LP)** for computing the fractional edge cover number to be defined next. Note that even though our weight function is defined to take values between 0 and 1, we do not need to add $\gamma(e) \leq 1$ as a constraint, because implicitly by the minimization itself the weight on an edge for an edge cover is never greater than 1. Also note that now the program above is an LP, which (in contrast to an ILP) can be solved in PTIME even if k is not fixed.

Definition 2.2. A *fractional edge cover* of a hypergraph $H = (V(H), E(H))$ is a function $\gamma : E(H) \rightarrow [0, 1]$ such that $V(H) = B(\gamma)$. The *fractional edge cover number* $\rho^*(H)$ of H is the minimum weight of all fractional edge covers of H. We write $supp(\gamma)$ to denote the *support* of γ, i.e., $supp(\gamma) := \{e \in E(H) \mid \gamma(e) > 0\}$.

We also extend the above definitions to subsets $S \subseteq V(H)$, i.e., an edge cover of S in H is a function $\lambda : E(H) \rightarrow \{0, 1\}$ such that $S \subseteq B(\lambda)$. If H is clear from the context, then we shall simply speak of "an edge cover of S" without explicitly mentioning H. The edge cover number $\rho(S)$ is then the minimum weight of all edge covers of S. The definitions for the fractional case are extended analogously.

Clearly, we have $\rho^*(H) \leq \rho(H)$ for every hypergraph H, and $\rho^*(H)$ can be much smaller than $\rho(H)$. However, below, we give an example, which is important for our proof of Theorem 3.2 and where $\rho^*(H)$ and $\rho(H)$ coincide.

Lemma 2.3. *Let K_{2n} be a clique of size $2n$. Then the equalities $\rho(K_{2n}) = \rho^*(K_{2n}) = n$ hold.*

Proof. Since we have to cover each vertex with weight ≥ 1, the total weight on the vertices of the graph is $\geq 2n$. As the weight of each edge adds to the weight of at most two vertices, we need at least weight n on the edges to achieve $\geq 2n$ weight on the vertices. However, we can use n edges each with weight 1 to cover $2n$ vertices. Hence, in total, we get $n \leq \rho^*(K_{2n}) \leq \rho(K_{2n}) \leq n$. □

2.3 HDs, GHDs, and FHDs

We now define tree decompositions and three types of hypergraph decompositions:

Definition 2.4. A ***tree decomposition*** **(TD)** of a hypergraph $H = (V(H), E(H))$ is a tuple $\langle T, (B_u)_{u \in N(T)} \rangle$, such that $T = \langle N(T), E(T) \rangle$ is a rooted tree and the following conditions hold:

(1) for each $e \in E(H)$, there is a node $u \in N(T)$ with $e \subseteq B_u$;
(2) for each $v \in V(H)$, the set $\{u \in N(T) \mid v \in B_u\}$ is connected.

Definition 2.5. A ***generalized hypertree decomposition*** **(GHD)** of a hypergraph $H = (V(H), E(H))$ is a tuple $\langle T, (B_u)_{u \in N(T)}, (\lambda)_{u \in N(T)} \rangle$, where $\langle T, (B_u)_{u \in N(T)} \rangle$ is a TD of H and, additionally, the following condition (3) holds:

(3) for each $u \in N(T)$, λ_u is a function $\lambda_u : E(H) \rightarrow \{0, 1\}$ with $B_u \subseteq B(\lambda_u)$.

Let us clarify some notational conventions used throughout this article. To avoid confusion, we will consequently refer to the elements in $V(H)$ as *vertices* (of the hypergraph) and to the elements in $N(T)$ as the *nodes* of T (of the decomposition). Now consider a decomposition $\mathcal{G}$ with tree structure T. For a node u in T, we write T_u to denote the subtree of T rooted at u. By slight abuse of notation, we will often write $u' \in T_u$ to denote that u' is a node in the subtree T_u of T. Moreover, we define $V(T_u) := \bigcup_{u' \in T_u} B_{u'}$ and, for a set $V' \subseteq V(H)$, we define $nodes(V') = \{u \in T \mid B_u \cap V' \neq \emptyset\}$. If we want to make explicit the decomposition $\mathcal{G}$, then we also write $nodes(V', \mathcal{G})$ synonymously with $nodes(V')$. By further overloading the *nodes* operator, we also write $nodes(T_u)$ or $nodes(T_u, \mathcal{G})$ to denote the nodes in a subtree T_u of T, i.e., $nodes(T_u) = nodes(T_u, \mathcal{G}) = \{v \mid v \in T_u\}$.

Definition 2.6. A ***hypertree decomposition*** **(HD)** of a hypergraph $H = (V(H), E(H))$ is a GHD, which in addition also satisfies the following condition (4):

(4) for each $u \in N(T)$, $V(T_u) \cap B(\lambda_u) \subseteq B_u$.

Definition 2.7. A ***fractional hypertree decomposition*** **(FHD)** [35] of a hypergraph $H = (V(H), E(H))$ is a tuple $\langle T, (B_u)_{u \in N(T)}, (\gamma)_{u \in N(T)} \rangle$, where $\langle T, (B_u)_{u \in N(T)} \rangle$ is a TD of H and, additionally, the following condition (3') holds:

(3') for each $u \in N(T)$, γ_u is a function $\gamma_u : E(H) \rightarrow [0,1]$ with $B_u \subseteq B(\gamma_u)$.

The width of a GHD, HD, or FHD is the maximum weight of the functions λ_u or γ_u, respectively, over all nodes u in T. Moreover, the generalized hypertree width, hypertree width, and fractional hypertree width of H (denoted $ghw(H)$, $hw(H)$, $fhw(H)$) is the minimum width over all GHDs, HDs, and FHDs of H, respectively. Alternatively, we could define the ghw of a TD $\left\langle T, (B_u)_{u \in N(T)} \right\rangle$ as $\max_{u \in N(T)} \rho(B_u)$. It is clear that the definitions lead to equivalent notions of ghw for a hypergraph. The same is true for the fhw of a TD, which corresponds to $\max_{u \in N(T)} \rho^*(B_u)$. Condition (2) is usually called the "connectedness condition," and condition (4) is referred to as "special condition" [31]. The set B_u is often referred to as the "bag" at node u. Note that, strictly speaking, only HDs require that the underlying tree T be rooted. For the sake of a uniform treatment, we assume that also the tree underlying a GHD or an FHD is rooted (with the understanding that the root is arbitrarily chosen).

We now recall two fundamental properties of the various notions of decompositions and width.

Lemma 2.8. *Let H be a hypergraph and let H' be a vertex induced subhypergraph of H, then $hw(H') \leq hw(H)$, $ghw(H') \leq ghw(H)$, and $fhw(H') \leq fhw(H)$ hold.*

Lemma 2.9. *Let H be a hypergraph. If there exists a vertex set $S \subseteq V(H)$ such that S is a clique in the primal graph, then every HD, GHD, or FHD of H has a node u such that $S \subseteq B_u$.*

Strictly speaking, Lemma 2.9 is a well-known property of tree decompositions—independently of the λ- or γ-label.

3 NP-HARDNESS

The main result in this section is the NP-hardness of Check($decomp, k$) with $decomp \in$ {GHD, FHD} and $k = 2$. The proof is rather technical, we therefore begin with an informal overview of the overall strategy.

The reduction is from 3SAT. We introduce the sets Y, Y' for the literals, sets $A = \{a_1, \dots, a_m\}$, $A' = \{a'_1, \dots, a'_m\}$, set S, and two special vertices $\{z_1, z_2\}$. The hypergraph H to be constructed consists of three main parts: two versions of the gadget in Figure 2 (which we will refer to as H_0 and H'_0, respectively) and a subhypergraph encoding the clauses of the 3SAT instance. The subhypergraph H_0 contains the vertices A, Y, S, z_1, z_2, while the subhypergraph H'_0 contains the vertices A', Y', S, z_1, z_2. The intended GHD/FHD consists of a long path that connects a decomposition of H_0 at its left end and a decomposition of H'_0 at its right end. A sketch of this decomposition is shown in Figure 1.

We assume that each of the bags $B_1, \dots, B_m$ on the long path contains a subset of $A \cup A'$, a subset Y_{sol} of $Y \cup Y'$ corresponding to the solution, S, and z_1, z_2. Each such bag should be covered by a pair of edges, such that one covers z_1 and there is a unique edge covering z_2 that it can be paired with and vice versa: Every edge covering z_1 covers all but a small subset of vertices of S, and there is a unique edge covering z_2 that fills this hole. Furthermore, every pair of edges covering $S \cup \{z_1, z_2\}$ covers exactly a subset $A^*_j = \{a'_1, \dots, a'_j, a_j, \dots, a_m\}$ of $A \cup A'$ for some j. By making a_j and a'_j adjacent for every j, we can make sure that the only way the decomposition can move from A on the left to A' on the right is if bag B_j contains exactly such a subset A^*_j of $A \cup A'$. Then, we make the following connection to the original 3SAT instance: There are exactly three pairs of edges covering $\{a'_1, \dots, a'_j, a_j, \dots, a_m\}$ corresponding to the three literals of the jth clause $(\ell_1 \vee \ell_2 \vee \ell_3)$ of the formula. Each such pair fully covers $Y \cup Y'$ except for the *negation* of one of the literals (i.e., $\overline{\ell_1}$, $\overline{\ell_2}$, or $\overline{\ell_3}$). If Y_{sol} satisfies the clause, then it *does not* contain one of $\overline{\ell_1}$, $\overline{\ell_2}$, or $\overline{\ell_3}$, so one of the three pairs can fully cover Y_{sol} and hence all of bag B_j. In the end, the bags $B_1, \dots, B_m$ verify that each of the m clauses of the formula is satisfied by Y_{sol}.

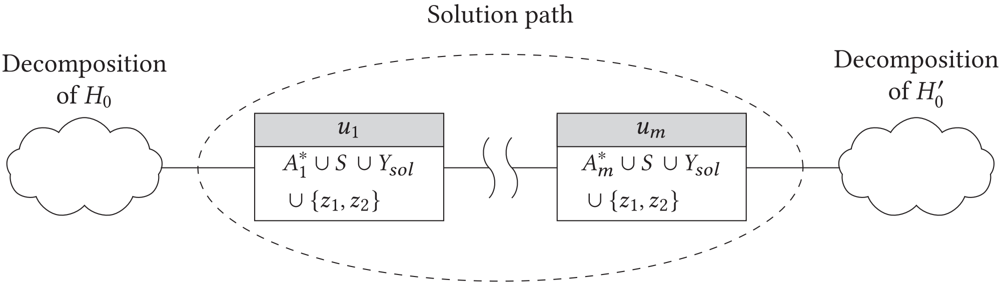
Solution path
Decomposition of H_0
Decomposition of H'_0
u_1
$A_1^* \cup S \cup Y_{sol} \cup \{z_1, z_2\}$
u_m
$A_m^* \cup S \cup Y_{sol} \cup \{z_1, z_2\}$

Fig. 1. Sketch of the intended GHD/FHD.

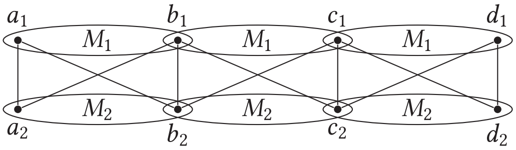

Fig. 2. Basic structure of H_0 in Lemma 3.1.

Our gadget construction makes sure that the decompositions of H_0 and H'_0 indeed contain the specified sets of vertices (see Lemma 3.1 for the details). Moreover, we have to ensure that the same subset Y_{sol} of $Y \cup Y'$ appears in every $B_1, \dots, B_j$ if we want to read out a satisfying assignment from the decomposition. In principle, it is possible that a y'_i appears or a y_i disappears as we move from B_1 to B_m. But there is an easy standard solution for this problem: Let us repeat the path $2n+1$ times, doing the full check of the m clauses $2n+1$ times. As there are only $2n$ possible changes of the y_i and y'_i vertices, one of the $2n+1$ copies is free of changes, and hence can be used to deduce a satisfying assignment.

We begin by defining the aforementioned gadget (Figure 2), which will play an integral part of this construction. Its crucial properties are stated below.

Lemma 3.1. *Let M_1, M_2 be disjoint sets and $M = M_1 \cup M_2$. Let $H = (V(H), E(H))$ be a hypergraph and $H_0 = (V_0, E_A \cup E_B \cup E_C)$ a subhypergraph of H with $V_0 = \{a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2\} \cup M$ and*

$$\begin{aligned} E_A &= \{\{a_1, b_1\} \cup M_1, \{a_2, b_2\} \cup M_2, \{a_1, b_2\}, \{a_2, b_1\}, \{a_1, a_2\}\}, \\ E_B &= \{\{b_1, c_1\} \cup M_1, \{b_2, c_2\} \cup M_2, \{b_1, c_2\}, \{b_2, c_1\}, \{b_1, b_2\}, \{c_1, c_2\}\}, \\ E_C &= \{\{c_1, d_1\} \cup M_1, \{c_2, d_2\} \cup M_2, \{c_1, d_2\}, \{c_2, d_1\}, \{d_1, d_2\}\}, \end{aligned}$$

where no element from the set $R = \{a_2, b_1, b_2, c_1, c_2, d_1, d_2\}$ occurs in any edge of $E(H) \setminus (E_A \cup E_B \cup E_C)$. Then, every FHD $\mathcal{F} = \langle T, (B_u)_{u \in T}, (\gamma_u)_{u \in T} \rangle$ of width ≤ 2 of H has nodes u_A, u_B, u_C s.t.:

- *$\{a_1, a_2, b_1, b_2\} \subseteq B_{u_A} \subseteq M \cup \{a_1, a_2, b_1, b_2\}$,*
- *$B_{u_B} = \{b_1, b_2, c_1, c_2\} \cup M$,*
- *$\{c_1, c_2, d_1, d_2\} \subseteq B_{u_C} \subseteq M \cup \{c_1, c_2, d_1, d_2\}$, and*
- *u_B is on the path from u_A to u_C.*

Proof. Consider an arbitrary FHD $\mathcal{F} = \langle T, (B_u)_{u \in T}, (\gamma_u)_{u \in T} \rangle$ of width ≤ 2 of H. Observe that a_1, a_2, b_1, and b_2 form a clique of size 4 in the primal graph. Hence, by Lemma 2.9, there is a node u_A in $\mathcal{F}$, such that $\{a_1, a_2, b_1, b_2\} \subseteq B_{u_A}$. It remains to show that also $B_{u_A} \subseteq M \cup \{a_1, a_2, b_1, b_2\}$ holds. To this end, we use a similar reasoning as in the proof of Lemma 2.3: to cover each vertex in $\{a_1, a_2, b_1, b_2\}$, we have to put weight ≥ 1 on each of these four vertices. By assumption, the only edges containing two out of these four vertices are the edges in $E_A \cup \{\{b_1, b_2\}\}$. All other edges in $E(H)$ contain at most one out of these four vertices. Hence, to cover $\{a_1, a_2, b_1, b_2\}$ with weight ≤ 2, we are only allowed to put non-zero weight on the edges in $E_A \cup \{\{b_1, b_2\}\}$. It follows that $B_{u_A} \subseteq M \cup \{a_1, a_2, b_1, b_2\}$ indeed holds.

Analogously, for the cliques b_1, b_2, c_1, c_2 and c_1, c_2, d_1, d_2, there must exist nodes u_B and u_C in $\mathcal{F}$ with $\{b_1, b_2, c_1, c_2\} \subseteq B_{u_B} \subseteq M \cup \{b_1, b_2, c_1, c_2\}$ and $\{c_1, c_2, d_1, d_2\} \subseteq B_{u_C} \subseteq M \cup \{c_1, c_2, d_1, d_2\}$.

It remains to show that u_B is on the path from u_A to u_C and $B_{u_B} = \{b_1, b_2, c_1, c_2\} \cup M$ holds. We first show that u_B is on the path between u_A and u_C. Suppose to the contrary that it is not. We distinguish three cases. First, assume that u_A is on the path between u_B and u_C. Then, by connectedness, $\{c_1, c_2\} \subseteq B_{u_A}$, which contradicts the property $B_{u_A} \subseteq M \cup \{a_1, a_2, b_1, b_2\}$ shown above. Second, assume u_C is on the path between u_A and u_B. In this case, we have $\{b_1, b_2\} \subseteq B_{u_C}$, which contradicts the property $B_{u_C} \subseteq M \cup \{c_1, c_2, d_1, d_2\}$ shown above. Third, suppose there is no path containing u_A, u_B, and u_C. Then there exists a node u such that removing u from T would put u_A, u_B, u_C in three different components. Node u is therefore on the path between any pair of the three nodes. Because u is on the path from u_A to u_B, $\{b_1, b_s\} \subseteq B_u$ by connectedness. Analogously, also $\{c_1, c_2\} \subseteq B_u$ because of the path from u_C to u_B. Then, by the same argument as in the beginning, we have $\{b_1, b_2, c_1, c_2\} \subseteq B_u \subseteq M \cup \{b_1, b_2, c_1, c_2\}$. Hence, u satisfies all the properties we have established for u_B and we can just consider the node u, which is on the path from u_A to u_C, to be our u_B.

We now show that also $B_{u_B} = \{b_1, b_2, c_1, c_2\} \cup M$ holds. Since we have already established $\{b_1, b_2, c_1, c_2\} \subseteq B_{u_B} \subseteq M \cup \{b_1, b_2, c_1, c_2\}$, it suffices to show $M \subseteq B_{u_B}$. First, let T'_a be the subgraph of T induced by $nodes(\{a_1, a_2\}, \mathcal{F})$ and let T'_d be the subgraph of T induced by $nodes(\{d_1, d_2\}, \mathcal{F})$. We show that each of the subgraphs T'_a and T'_d is connected (i.e., a subtree of T) and that the two subtrees are disjoint. The connectedness is immediate: By the connectedness condition, each of $nodes(\{a_1\}, \mathcal{F})$, $nodes(\{a_2\}, \mathcal{F})$, $nodes(\{d_1\}, \mathcal{F})$, and $nodes(\{d_2\}, \mathcal{F})$ is connected. Moreover, since H contains an edge $\{a_1, a_2\}$ (respectively, $\{d_1, d_2\}$), the two subtrees induced by $nodes(\{a_1\}, \mathcal{F})$, $nodes(\{a_2\}, \mathcal{F})$ (respectively, $nodes(\{d_1\}, \mathcal{F})$, $nodes(\{d_2\}, \mathcal{F})$) must be connected, hence T'_a and T'_d are subtrees of T. It remains to show that T'_a and T'_d are disjoint.

Clearly, $u_A \in T'_a$ and $u_C \in T'_d$. We have established above that u_B is on a path from u_A to u_C. Also, $B_{u_B} \subseteq M \cup \{b_1, b_2, c_1, c_2\}$ and therefore u_B is in neither T'_a nor T'_d. Because both are subtrees of the tree T, there is only one path connecting them. But u_B is on that path and in neither subtree, i.e., T'_a and T'_d are disjoint.

As every edge must be covered, there are nodes in T'_a that cover $\{a_1, b_1\} \cup M_1$ and $\{a_2, b_2\} \cup M_2$, respectively. Hence, the subtree T'_a covers $M = M_1 \cup M_2$, i.e., $M \subseteq \bigcup_{u \in T'_a} B_u$. Likewise, T'_d covers M. Since both subtrees are disjoint and u_B is on the path between them, by the connectedness condition, we have $M \subseteq B_{u_B}$. □

Theorem 3.2. *The* Check*(decomp, k) problem is* NP*-complete for decomp* $\in$ {*GHD, FHD*} *and* $k = 2$.

Proof. The problem is clearly in NP: Guess a tree decomposition and check in polynomial time for each node u whether $\rho(B_u) \leq 2$ or $\rho^*(B_u) \leq 2$, respectively, holds. The NP-hardness is proved by a reduction from 3SAT. Before presenting this reduction, we first introduce some useful notation.

Notation. For $i, j \geq 1$, we denote $\{1, \ldots, i\} \times \{1, \ldots, j\}$ by $[i; j]$. For each $p \in [i; j]$, we denote by $p \oplus 1$ ($p \ominus 1$) the successor (predecessor) of p in the usual lexicographic order on pairs, that is, the order $(1, 1), \ldots, (1, j)$, $(2, 1), \ldots, (i, 1), \ldots, (i, j)$. We refer to the first element $(1, 1)$ as min and to the last element (i, j) as max. We denote by $[i;j]^-$ the set $[i;j] \setminus \{\max\}$, i.e., $[i;j]$ without the last element.

Now let $\varphi = \bigwedge_{j=1}^{m}(L_j^1 \vee L_j^2 \vee L_j^3)$ be an arbitrary instance of 3SAT with m clauses and variables $x_1, \ldots, x_n$. From this, we will construct a hypergraph $H = (V(H), E(H))$, which consists of two copies H_0, H_0' of the (sub-)hypergraph H_0 of Lemma 3.1 plus additional edges connecting H_0 and H_0'. We use the sets $Y = \{y_1, \ldots, y_n\}$ and $Y' = \{y_1', \ldots, y_n'\}$ to encode the truth values of the variables of φ. We denote by Y_ℓ (Y_ℓ') the set $Y \setminus \{y_\ell\}$ $(Y' \setminus \{y_\ell'\})$. Furthermore, we use the sets $A = \{a_p \mid p \in [2n+3;m]\}$ and $A' = \{a_p' \mid p \in [2n+3;m]\}$, and we define the following subsets of A and A', respectively:

$$A_p = \{a_{\min}, \ldots, a_p\}, \qquad \overline{A_p} = \{a_p, \ldots, a_{\max}\},$$
$$A_p' = \{a_{\min}', \ldots, a_p'\}, \qquad \overline{A_p'} = \{a_p', \ldots, a_{\max}'\}.$$

In addition, we will use another set S of elements that controls and restricts the ways in which edges are combined in a possible FHD or GHD. Such a decomposition will have, implied by Lemma 3.1, two nodes u_B and u_B' such that $S \subseteq B_{u_B}$ and $S \subseteq B_{u_B'}$. From this, we will reason on the path connecting u_B and u_B'.

The concrete set S used in our construction of H is obtained as follows: Let $Q = [2n+3;m] \cup \{(0,1), (0,0), (1,0)\}$, hence, Q is an extension of the set $[2n+3;m]$ with special elements $(0,1), (0,0), (1,0)$. Then, we define the set S as $S = Q \times \{1,2,3\}$.

The elements in S are pairs, which we denote as $(q \mid k)$. The values $q \in Q$ are themselves pairs of integers (i, j). Intuitively, q indicates the position of a node on the "long" path π in the desired FHD or GHD. The integer k refers to a literal in the jth clause. We will write the wildcard $*$ to indicate that a component in some element of S can take an arbitrary value. For example, $(\min \mid *)$ denotes the set of tuples $(q \mid k)$ where $q = \min = (1,1)$ and k can take an arbitrary value in $\{1,2,3\}$. We will denote by S_p the set $(p \mid *)$. For instance, $(\min \mid *)$ will be denoted as $S_{\min}$. Further, for $p \in [2n+3;m]$ and $k \in \{1,2,3\}$, we define singletons $S_p^k = \{(p \mid k)\}$. **Problem reduction.** Let

$\varphi = \bigwedge_{j=1}^{m}(L_j^1 \vee L_j^2 \vee L_j^3)$ be an arbitrary instance of 3SAT with m clauses and variables $x_1, \ldots, x_n$. From this, we construct a hypergraph $H = (V(H), E(H))$, that is, an instance of Check(*decomp*, k) with *decomp* $\in$ {GHD, FHD} and $k = 2$.

We start by defining the vertex set $V(H)$:

$$V(H) = S \cup A \cup A' \cup Y \cup Y' \cup \{z_1, z_2\} \cup$$
$$\{a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2, a_1', a_2', b_1', b_2', c_1', c_2', d_1', d_2'\}.$$

The edges of H are defined in three steps. First, we take two copies of the subhypergraph H_0 used in Lemma 3.1:

- Let $H_0 = (V_0, E_0)$ be the hypergraph of Lemma 3.1 with $V_0 = \{a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2\} \cup M_1 \cup M_2$ and $E_0 = E_A \cup E_B \cup E_C$, where we set $M_1 = S \setminus S_{(0,1)} \cup \{z_1\}$ and $M_2 = Y \cup S_{(0,1)} \cup \{z_2\}$.
- Let $H_0' = (V_0', E_0')$ be the corresponding hypergraph, with $V_0' = \{a_1', a_2', b_1', b_2', c_1', c_2', d_1', d_2'\} \cup M_1' \cup M_2'$ and E_A', E_B', E_C' are the primed versions of the edge sets $M_1' = S \setminus S_{(1,0)} \cup \{z_1\}$ and $M_2' = Y' \cup S_{(1,0)} \cup \{z_2\}$.

In the second step, we define the edges that (as we will see) enforce the existence of a "long" path π between the nodes covering H_0 and the nodes covering H_0' in any FHD of width ≤ 2.

- $e_p = A_p' \cup \overline{A_p}$, for $p \in [2n+3;m]^-$,
- $e_{y_i} = \{y_i, y_i'\}$, for $1 \leq i \leq n$,

- For $p = (i, j) \in [2n + 3; m]^{-}$ and $k \in \{1, 2, 3\}$:

$$e_p^{k,0} = \begin{cases} \overline{A_p} \cup (S \setminus S_p^k) \cup Y \cup \{z_1\} & \text{if } L_j^k = x_\ell \\ \overline{A_p} \cup (S \setminus S_p^k) \cup Y_\ell \cup \{z_1\} & \text{if } L_j^k = \neg x_\ell, \end{cases}$$

$$e_p^{k,1} = \begin{cases} A'_p \cup S_p^k \cup Y'_\ell \cup \{z_2\} & \text{if } L_j^k = x_\ell \\ A'_p \cup S_p^k \cup Y' \cup \{z_2\} & \text{if } L_j^k = \neg x_\ell. \end{cases}$$

Finally, we need edges that connect H_0 and H'_0 with the above edges covered by the nodes of the "long" path π in a GHD or FHD:

- $e^0_{(0,0)} = \{a_1\} \cup A \cup S \setminus S_{(0,0)} \cup Y \cup \{z_1\}$,
- $e^1_{(0,0)} = S_{(0,0)} \cup Y' \cup \{z_2\}$,
- $e^0_{\max} = S \setminus S_{\max} \cup Y \cup \{z_1\}$,
- $e^1_{\max} = \{a'_1\} \cup A' \cup S_{\max} \cup Y' \cup \{z_2\}$.

This concludes the construction of the hypergraph H. Before we prove the correctness of the problem reduction, we give an example that will help to illustrate the intuition underlying this construction.

Example 3.3. Suppose that an instance of 3SAT is given by the propositional formula $\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3)$, i.e.: We have $n = 3$ variables and $m = 2$ clauses. From this, we construct a hypergraph $H = (V(H), E(H))$. First, we instantiate the sets Q, A, A', S, Y, and Y' from our problem reduction.

$$\begin{aligned} A &= \{a_{(1,1)}, a_{(1,2)}, a_{(2,1)}, a_{(2,2)}, \dots, a_{(9,1)}, a_{(9,2)}\}, \\ A' &= \{a'_{(1,1)}, a'_{(1,2)}, a'_{(2,1)}, a'_{(2,2)}, \dots, a'_{(9,1)}, a'_{(9,2)}\}, \\ Q &= \{(1,1), (1,2), (2,1), (2,2), \dots, (9,1), (9,2)\} \cup \{(0,1), (0,0), (1,0)\}, \\ S &= Q \times \{1, 2, 3\}, \\ Y &= \{y_1, y_2, y_3\}, \\ Y' &= \{y'_1, y'_2, y'_3\}. \end{aligned}$$

According to our problem reduction, the set $V(H)$ of vertices of H is

$$\begin{aligned} V(H) = S \;\cup\; A \;\cup\; A' \;\cup\; Y \;\cup\; Y' \;\cup\; \{z_1, z_2\} \cup \\ \{a_1, a_2, b_1, b_2, c_1, c_2, d_1, d_2\} \cup \{a'_1, a'_2, b'_1, b'_2, c'_1, c'_2, d'_1, d'_2\}. \end{aligned}$$

The set $E(H)$ of edges of H is defined in several steps. First, the edges in H_0 and H'_0 are defined: We thus have the subsets $E_A, E_B, E_C, E'_A, E'_B, E'_C \subseteq E(H)$, whose definition is based on the sets $M_1 = S \setminus S_{(0,1)} \cup \{z_1\}$, $M_2 = Y \cup S_{(0,1)} \cup \{z_2\}$, $M'_1 = S \setminus S_{(1,0)} \cup \{z_1\}$, and $M'_2 = Y' \cup S_{(1,0)} \cup \{z_2\}$. The definition of the edges

$$\begin{aligned} e_p &= A'_p \cup \overline{A_p} && \text{for } p \in \{(1,1), (1,2), \dots (8,1), (8,2), (9,1)\}, \\ e_{y_i} &= \{y_i, y'_i\} && \text{for } 1 \le i \le 3, \\ e^0_{(0,0)} &= \{a_1\} \cup A \cup S \setminus S_{(0,0)} \cup Y \cup \{z_1\}, \\ e^1_{(0,0)} &= S_{(0,0)} \cup Y' \cup \{z_2\}, \\ e^0_{(9,2)} &= S \setminus S_{(9,2)} \cup Y \cup \{z_1\}, \text{ and} \\ e^1_{(9,2)} &= \{a'_1\} \cup A' \cup S_{(9,2)} \cup Y' \cup \{z_2\} \end{aligned}$$

is straightforward. We concentrate on the edges $e_p^{k,0}$ and $e_p^{k,1}$ for $p \in \{(1,1),(1,2),\dots(8,1),(8,2),(9,1)\}$ and $k \in \{1,2,3\}$. These edges play the key role for covering the bags of the nodes along the "long" path π in any FHD or GHD of H. This path can be thought of as being structured in nine blocks. Consider an arbitrary $i \in \{1,\dots,9\}$. Then $e_{(i,1)}^{k,0}$ and $e_{(i,1)}^{k,1}$ encode the kth literal of the first clause and $e_{(i,2)}^{k,0}$ and $e_{(i,2)}^{k,1}$ encode the kth literal of the second clause (the latter is only defined for $i \leq 8$). These edges are defined as follows: The edges $e_{(i,1)}^{1,0}$ and $e_{(i,1)}^{1,1}$ encode the first literal of the first clause, i.e., the positive literal x_1. We thus have

$$\begin{array}{rcl} e_{(i,1)}^{1,0} & = & \overline{A_{(i,1)}} \cup (S \setminus S_{(i,1)}^1) \cup \{y_1,y_2,y_3\} \cup \{z_1\} \text{ and} \\ e_{(i,1)}^{1,1} & = & A'_{(i,1)} \cup S_{(i,1)}^1 \cup \{y'_2,y'_3\} \cup \{z_2\}. \end{array}$$

The edges $e_{(i,1)}^{2,0}$ and $e_{(i,1)}^{2,1}$ encode the second literal of the first clause, i.e., the negative literal $\neg x_2$. Likewise, $e_{(i,1)}^{3,0}$ and $e_{(i,1)}^{3,1}$ encode the third literal of the first clause, i.e., the positive literal x_3. Hence,

$$\begin{array}{rcl} e_{(i,1)}^{2,0} & = & \overline{A_{(i,1)}} \cup (S \setminus S_{(i,1)}^2) \cup \{y_1,y_3\} \cup \{z_1\}, \\ e_{(i,1)}^{2,1} & = & A'_{(i,1)} \cup S_{(i,1)}^2 \cup \{y'_1,y'_2,y'_3\} \cup \{z_2\}, \\ e_{(i,1)}^{3,0} & = & \overline{A_{(i,1)}} \cup (S \setminus S_{(i,1)}^3) \cup \{y_1,y_2,y_3\} \cup \{z_1\}, \text{ and} \\ e_{(i,1)}^{3,1} & = & A'_{(i,1)} \cup S_{(i,1)}^3 \cup \{y'_1,y'_2\} \cup \{z_2\}. \end{array}$$

Analogously, the edges $e_{(i,2)}^{1,0}$ and $e_{(i,2)}^{1,1}$ (encoding the first literal of the second clause, i.e., $\neg x_1$), the edges $e_{(i,2)}^{2,0}$ and $e_{(i,2)}^{2,1}$ (encoding the second literal of the second clause, i.e., x_2), and the edges $e_{(i,2)}^{3,0}$ and $e_{(i,2)}^{3,1}$ (encoding the third literal of the second clause, i.e., $\neg x_3$) are defined as follows:

$$\begin{array}{rcl} e_{(i,2)}^{1,0} & = & \overline{A_{(i,2)}} \cup (S \setminus S_{(i,2)}^1) \cup \{y_2,y_3\} \cup \{z_1\}, \\ e_{(i,2)}^{1,1} & = & A'_{(i,2)} \cup S_{(i,2)}^1 \cup \{y'_1,y'_2,y'_3\} \cup \{z_2\}, \\ e_{(i,2)}^{2,0} & = & \overline{A_{(i,2)}} \cup (S \setminus S_{(i,2)}^2) \cup \{y_1,y_2,y_3\} \cup \{z_1\}, \\ e_{(i,2)}^{2,1} & = & A'_{(i,2)} \cup S_{(i,2)}^2 \cup \{y'_1,y'_3\} \cup \{z_2\}, \\ e_{(i,2)}^{3,0} & = & \overline{A_{(i,2)}} \cup (S \setminus S_{(i,2)}^3) \cup \{y_1,y_2\} \cup \{z_1\}, \text{ and} \\ e_{(i,2)}^{3,1} & = & A'_{(i,2)} \cup S_{(i,2)}^3 \cup \{y'_1,y'_2,y'_3\} \cup \{z_2\}. \end{array}$$

The crucial property of these pairs of edges $e_{(i,j)}^{k,0}$ and $e_{(i,j)}^{k,1}$ is that they together encode the kth literal of the jth clause in the following way: If the literal is of the form x_ℓ (respectively, of the form $\neg x_\ell$), then $e_{(i,j)}^{k,0} \cup e_{(i,j)}^{k,1}$ covers all of $Y \cup Y'$ except for y'_ℓ (respectively, except for y_ℓ).

Formula φ in this example is clearly satisfiable, e.g., by the truth assignment σ with $\sigma(x_1) =$ true and $\sigma(x_2) = \sigma(x_3) =$ false. Hence, for the problem reduction to be correct, there must exist a GHD (and thus also an FHD) of width 2 of H. In Figure 3, the tree structure T plus the bags $(B_t)_{t \in T}$ of such a GHD is displayed. Moreover, in Table 1, the precise definition of B_u and λ_u of every node $u \in T$ is given: In the column labelled B_u, the set of vertices contained in B_u for each node $u \in T$ is shown. In the column labelled λ_u, the two edges with weight 1 are shown. For the row with label $u_{p \in [2n+3;m]^-}$, the entry in the last column is $e_p^{k_p,0}, e_p^{k_p,1}$. By this, we mean that, for every p, an appropriate value $k_p \in \{1,2,3\}$ has to be determined. It will be explained below how to find an appropriate value k_p for each p. The set Z in the bags of this GHD is defined as

$Z = \{y_i \mid \sigma(x_i) = \text{true}\} \cup \{y'_i \mid \sigma(x_i) = \text{false}\}$. In this example, for the chosen truth assignment σ, we thus have $Z = \{y_1, y'_2, y'_3\}$. The bags B_t and the edge covers λ_t for each $t \in T$ are explained below.

The nodes u_C, u_B, u_A to cover the edges of the subhypergraph H_0 and the nodes u'_A, u'_B, u'_C to cover the edges of the subhypergraph H'_0 are clear by Lemma 3.1. The purpose of the nodes $u_{\min \ominus 1}$ and $u_{\max}$ is mainly to make sure that each edge $\{y_i, y'_i\}$ is covered by some bag. Recall that the set Z contains exactly one of y_i and y'_i for every i. Hence, the node $u_{\min \ominus 1}$ (respectively, $u_{\max}$) covers each edge $\{y_i, y'_i\}$, such that $y'_i \in Z$ (respectively, $y_i \in Z$).

We now have a closer look at the nodes $u_{(1,1)}$ to $u_{(9,1)}$ on the "long" path π. More precisely, let us look at the nodes $u_{(i,1)}$ and $u_{(i,2)}$ for some $i \in \{1, \ldots, 8\}$, i.e., the "ith block." It will turn out that the bags at these nodes can be covered by edges from H because φ is satisfiable. Indeed, our choice of $\lambda_{u_{(i,1)}}$ and $\lambda_{u_{(i,2)}}$ is guided by the literals satisfied by the truth assignment σ, namely: For $\lambda_{u_{(i,j)}}$, we have to choose some k_j, such that the k_jth literal in the jth clause is true in σ. For instance, we may define $\lambda_{u_{(i,1)}}$ and $\lambda_{u_{(i,2)}}$ as follows:

$$\lambda_{u_{(i,1)}} = \left\{e^{1,0}_{(i,1)}, e^{1,1}_{(i,1)}\right\}, \qquad \lambda_{u_{(i,2)}} = \left\{e^{3,0}_{(i,2)}, e^{3,1}_{(i,2)}\right\}.$$

The covers $\lambda_{u_{(i,1)}}$ and $\lambda_{u_{(i,2)}}$ were chosen because the first literal of the first clause and the third literal of the second clause are true in σ. Now let us verify that $\lambda_{u_{(i,1)}}$ and $\lambda_{u_{(i,2)}}$ are indeed covers of $B_{u_{(i,1)}}$ and $B_{u_{(i,2)}}$, respectively. By the definition of the edges $e^{k,0}_{(i,j)}, e^{k,1}_{(i,j)}$ for $j \in \{1,2\}$ and $k \in \{1,2,3\}$, it is immediate that $e^{k,0}_{(i,j)} \cup e^{k,1}_{(i,j)}$ covers $\overline{A_{(i,j)}} \cup A'_{(i,j)} \cup S \cup \{z_1, z_2\}$. The only non-trivial question is if $\lambda_{u_{(i,j)}}$ also covers Z. Recall that by definition, $(e^{1,0}_{(i,1)} \cup e^{1,1}_{(i,1)}) \supseteq (Y \cup Y') \setminus \{y'_1\}$. Our truth assignment σ sets $\sigma(x_1) = \text{true}$. Hence, by our definition of Z, we have $y_1 \in Z$ and $y'_1 \notin Z$. This means that $e^{1,0}_{(i,1)} \cup e^{1,1}_{(i,1)}$ indeed covers Z and, hence, all of $B_{u_{(i,1)}}$. Note that we could have also chosen $\lambda_{u_{(i,1)}} = \{e^{2,0}_{(i,1)}, e^{2,1}_{(i,1)}\}$, since also the second literal of the first clause (i.e., $\neg x_2$) is true in σ. In this case, we would have $(e^{2,0}_{(i,1)} \cup e^{2,1}_{(i,1)}) \supseteq (Y \cup Y') \setminus \{y_2\}$ and Z indeed does not contain y_2. Conversely, setting $\lambda_{u_{(i,1)}} = \{e^{3,0}_{(i,1)}, e^{3,1}_{(i,1)}\}$ would fail, because in this case, $y'_3 \notin (e^{3,0}_{(i,1)} \cup e^{3,1}_{(i,1)})$, since x_3 occurs positively in the first clause. However, we have $y'_3 \in Z$ by definition of Z, because $\sigma(x_3) = \text{false}$ holds.

Checking that $\lambda_{u_{(i,2)}}$ as defined above covers Z is done analogously. Note that in the second clause, only the third literal is satisfied by σ. Hence, setting $\lambda_{u_{(i,2)}} = \{e^{3,0}_{(i,2)}, e^{3,1}_{(i,2)}\}$ is the only option to cover $B_{u_{(i,2)}}$ (in particular, to cover Z). Finally, note that σ as defined above is not the only satisfying truth assignment of φ. For instance, we could have chosen $\sigma(x_1) = \sigma(x_2) = \sigma(x_3) = \text{true}$. In this case, we would define $Z = \{y_1, y_2, y_3\}$ and the covers $\lambda_{u_{(i,j)}}$ would have to be chosen according to an arbitrary choice of one literal per clause that is satisfied by this assignment σ. ◇

To prove the correctness of our problem reduction, we have to show the two equivalences: first, that $ghw(H) \leq 2$ if and only if φ is satisfiable and, second, that $fhw(H) \leq 2$ if and only if φ is satisfiable. We prove the two directions of these equivalences separately.

Proof of the "if"-direction. First assume that φ is satisfiable. It suffices to show that then H has a GHD of width ≤2, because $fhw(H) \leq ghw(H)$ holds. Let σ be a satisfying truth assignment. Let us fix for each $j \leq m$, some $k_j \in \{1,2,3\}$ such that $\sigma(L^{k_j}_j) = 1$. By ℓ_j, we denote the index of the variable in the literal $L^{k_j}_j$, that is, $L^{k_j}_j = x_{\ell_j}$ or $L^{k_j}_j = \neg x_{\ell_j}$. For $p = (i,j)$, let k_p refer to k_j and let $L^{k_p}_p$ refer to $L^{k_j}_j$. Finally, we define Z as $Z = \{y_i \mid \sigma(x_i) = 1\} \cup \{y'_i \mid \sigma(x_i) = 0\}$.

A GHD $\mathcal{G} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T}\rangle$ of width 2 for H is constructed as follows: T is a path u_C, u_B, u_A, $u_{\min \ominus 1}$, $u_{\min}, \ldots$, $u_{\max}$, u'_A, u'_B, u'_C. The construction is illustrated in Figure 3. The precise

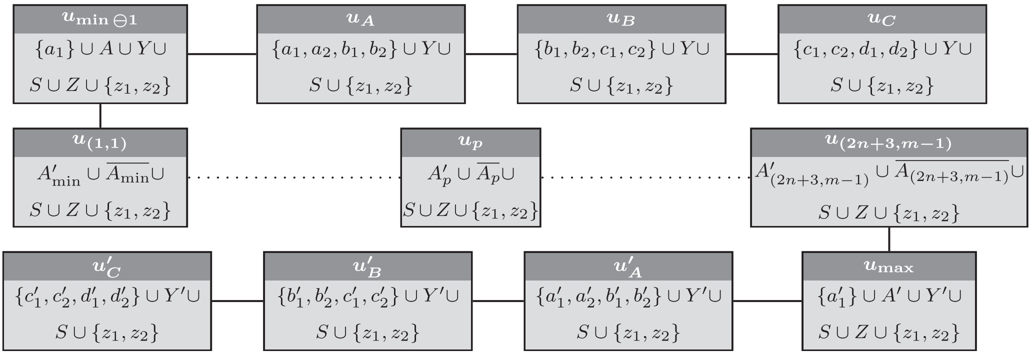
$u_{\min \ominus 1}$
$\{a_1\} \cup A \cup Y \cup$
$S \cup Z \cup \{z_1, z_2\}$
u_A
$\{a_1, a_2, b_1, b_2\} \cup Y \cup$
$S \cup \{z_1, z_2\}$
u_B
$\{b_1, b_2, c_1, c_2\} \cup Y \cup$
$S \cup \{z_1, z_2\}$
u_C
$\{c_1, c_2, d_1, d_2\} \cup Y \cup$
$S \cup \{z_1, z_2\}$
$u_{(1,1)}$
$A'_{\min} \cup \overline{A_{\min}} \cup$
$S \cup Z \cup \{z_1, z_2\}$
u_p
$A'_p \cup \overline{A_p} \cup$
$S \cup Z \cup \{z_1, z_2\}$
$u_{(2n+3,m-1)}$
$A'_{(2n+3,m-1)} \cup \overline{A_{(2n+3,m-1)}} \cup$
$S \cup Z \cup \{z_1, z_2\}$
u'_C
$\{c'_1, c'_2, d'_1, d'_2\} \cup Y' \cup$
$S \cup \{z_1, z_2\}$
u'_B
$\{b'_1, b'_2, c'_1, c'_2\} \cup Y' \cup$
$S \cup \{z_1, z_2\}$
u'_A
$\{a'_1, a'_2, b'_1, b'_2\} \cup Y' \cup$
$S \cup \{z_1, z_2\}$
$u_{\max}$
$\{a'_1\} \cup A' \cup Y' \cup$
$S \cup Z \cup \{z_1, z_2\}$

Fig. 3. Intended path of the GHD of hypergraph H in the proof of Theorem 3.2.

Table 1. Definition of B_u and λ_u for GHD of H

$u \in T$	B_u	λ_u
u_C	$\{d_1, d_2, c_1, c_2\} \cup Y \cup S \cup \{z_1, z_2\}$	$\{c_1, d_1\} \cup M_1$, $\{c_2, d_2\} \cup M_2$
u_B	$\{c_1, c_2, b_1, b_2\} \cup Y \cup S \cup \{z_1, z_2\}$	$\{b_1, c_1\} \cup M_1$, $\{b_2, c_2\} \cup M_2$
u_A	$\{b_1, b_2, a_1, a_2\} \cup Y \cup S \cup \{z_1, z_2\}$	$\{a_1, b_1\} \cup M_1$, $\{a_2, b_2\} \cup M_2$
$u_{\min \ominus 1}$	$\{a_1\} \cup A \cup Y \cup S \cup Z \cup \{z_1, z_2\}$	$e^0_{(0,0)}, e^1_{(0,0)}$
$u_{p \in [2n+3;m]^-}$	$A'_p \cup \overline{A_p} \cup S \cup Z \cup \{z_1, z_2\}$	$e_p^{k_p,0}, e_p^{k_p,1}$
$u_{\max}$	$\{a'_1\} \cup A' \cup Y' \cup S \cup Z \cup \{z_1, z_2\}$	$e^0_{\max}, e^1_{\max}$
u'_A	$\{a'_1, a'_2, b'_1, b'_2\} \cup Y' \cup S \cup \{z_1, z_2\}$	$\{a'_1, b'_1\} \cup M'_1$, $\{a'_2, b'_2\} \cup M'_2$
u'_B	$\{b'_1, b'_2, c'_1, c'_2\} \cup Y' \cup S \cup \{z_1, z_2\}$	$\{b'_1, c'_1\} \cup M'_1$, $\{b'_2, c'_2\} \cup M'_2$
u'_C	$\{c'_1, c'_2, d'_1, d'_2\} \cup Y' \cup S \cup \{z_1, z_2\}$	$\{c'_1, d'_1\} \cup M'_1$, $\{c'_2, d'_2\} \cup M'_2$

definition of B_u and λ_u is given in Table 1. Clearly, the GHD has width ≤2. We now show that $\mathcal{G}$ is indeed a GHD of H:

(1) For each edge $e \in E$, there is a node $u \in T$, such that $e \subseteq B_u$:
- $\forall e \in E_X : e \subseteq B_{u_X}$ for all $X \in \{A, B, C\}$,
- $\forall e' \in E'_X : e' \subseteq B_{u'_X}$ for all $X \in \{A, B, C\}$,
- $e_p \subseteq B_{u_p}$ for $p \in [2n+3; m]^-$,
- $e_{y_i} \subseteq B_{u_{\min \ominus 1}}$ (if $y'_i \in Z$) or $e_{y_i} \subseteq B_{u_{\max}}$ (if $y_i \in Z$), respectively,
- $e_p^{k,0} \subseteq B_{u_{\min \ominus 1}}$ for $p \in [2n+3; m]^-$,
- $e_p^{k,1} \subseteq B_{u_{\max}}$ for $p \in [2n+3; m]^-$,
- $e^0_{(0,0)} \subseteq B_{u_{\min \ominus 1}}$, $e^1_{(0,0)} \subseteq B_{u_{\max}}$,
- $e^0_{\max} \subseteq B_{u_{\min \ominus 1}}$ and $e^1_{\max} \subseteq B_{u_{\max}}$.

All of the above inclusions can be verified in Table 1.

(2) For each vertex $v \in V$, the set $\{u \in T \mid v \in B_u\}$ induces a connected subtree of T, which is easy to verify in Table 1.

(3) For each $u \in T$, $B_u \subseteq B(\lambda_u)$: The only inclusion that cannot be easily verified in Table 1 is $B_{u_p} \subseteq B(\lambda_{u_p})$. In fact, this is the only place in the proof where we make use of the assumption that φ is satisfiable. First, notice that the set $A'_p \cup \overline{A_p} \cup S \cup \{z_1, z_2\}$ is clearly a subset of $B(\lambda_{u_p})$. It remains to show that $Z \subseteq B(\lambda_{u_p})$ holds for arbitrary $p \in [2n+3; m]^-$. We show this property by a case distinction on the form of $L_p^{k_p}$.

Case (1): First, assume that $L_p^{k_p} = x_{\ell_j}$ holds. Then $\sigma(x_{\ell_j}) = 1$ and, therefore, $y'_{\ell_j} \notin Z$. But, by definition of $e_p^{k_p,0}$ and $e_p^{k_p,1}$, vertex y'_{ℓ_j} is the only element of $Y \cup Y'$ not contained in $B(\lambda_{u_p})$. Since $Z \subseteq (Y \cup Y')$ and $y'_{\ell_j} \notin Z$, we have that $Z \subseteq B(\lambda_{u_p})$.

Case (2): Now assume that $L_p^{k_p} = \neg x_{\ell_j}$ holds. Then $\sigma(x_{\ell_j}) = 0$ and, therefore, $y_{\ell_j} \notin Z$. But, by definition of $e_p^{k_p,0}$ and $e_p^{k_p,1}$, vertex y_{ℓ_j} is the only element of $Y \cup Y'$ not contained in $B(\lambda_{u_p})$. Since $Z \subseteq (Y \cup Y')$ and $y_{\ell_j} \notin Z$, we have that $Z \subseteq B(\lambda_{u_p})$.

Two crucial lemmas. Before we prove the "only if'-direction, we define the notion of complementary edges and state two important lemmas related to this notion.

Definition 3.4. Let e and e' be two edges from the hypergraph H as defined before. We say e' is the *complementary* edge of e (or, simply, e, e' are complementary edges) whenever

- $e \cap S = S \setminus S'$ for some $S' \subseteq S$ and
- $e' \cap S = S'$.

Observe that for every edge in our construction that covers $S \setminus S'$ for some $S' \subseteq S$ there is a complementary edge that covers S', for example, $e_p^{k,0}$ and $e_p^{k,1}$, $e^0_{(0,0)}$, and $e^1_{(0,0)}$, and so on. In particular, there is no edge that covers S completely. Moreover, consider arbitrary subsets S_1, S_2 of S, s.t. (syntactically) $S \setminus S_i$ is part of the definition of e_i for some $e_i \in E(H)$ with $i \in \{1, 2\}$. Then S_1 and S_2 are disjoint.

We now present two lemmas needed for the "only if"-direction.

Lemma 3.5. *Let $\mathcal{F} = \langle T, (B_u)_{u\in T}, (\gamma_u)_{u\in T}\rangle$ be an FHD of width ≤ 2 of the hypergraph H constructed above. For every node u with $S \cup \{z_1, z_2\} \subseteq B_u$ and every pair e, e' of complementary edges, it holds that $\gamma_u(e) = \gamma_u(e')$.*

Proof. First, we try to cover z_1 and z_2. For z_1, we have to put total weight 1 on the edges in E^0, and to cover z_2, we have to put total weight 1 on the edges in E^1, where

$$\begin{aligned} E^0 = {} & \left\{e_p^{k,0} \mid p \in [2n+3; m]^- \text{ and } 1 \leq k \leq 3\right\} \cup \\ & \left\{e^0_{(0,0)}, e^0_{\max}\right\} \cup \\ & \{\{a_1, b_1\} \cup M_1, \{b_1, c_1\} \cup M_1, \{c_1, d_1\} \cup M_1\} \cup \\ & \{\{a'_1, b'_1\} \cup M'_1, \{b'_1, c'_1\} \cup M'_1, \{c'_1, d'_1\} \cup M'_1\}, \\ E^1 = {} & \left\{e_p^{k,1} \mid p \in [2n+3; m]^- \text{ and } 1 \leq k \leq 3\right\} \cup \\ & \left\{e^1_{(0,0)}, e^1_{\max}\right\} \cup \\ & \{\{a_2, b_2\} \cup M_2, \{b_2, c_2\} \cup M_2, \{c_2, d_2\} \cup M_2\} \cup \\ & \{\{a'_2, b'_2\} \cup M'_2, \{b'_2, c'_2\} \cup M'_2, \{c'_2, d'_2\} \cup M'_2\}. \end{aligned}$$

To also cover S with weight 1, we are only allowed to assign weights to the above edges. Let e_i^0 be an arbitrary edge in E^0 with $\gamma_u(e_i^0) = w_i > 0$. Then there exists a subset S_i of S with $S_i \neq \emptyset$, s.t. $e_i^0 \cap S = S \setminus S_i$. Still, we need to put weight 1 on the vertices in S_i. To do so, we can put at most weight $1 - w_i$ on the edges in $E^0 \setminus \{e_i^0\}$, which covers S_i with weight at most $1 - w_i$. Hence, the edges in E^1 have to put weight $\geq w_i$ on S_i. The only edge in E^1 that intersects S_i is the complementary edge e_i^1 of e_i^0. Hence, we have to set $\gamma_u(e_i^1) \geq w_i$. In other words, we have to set $\gamma_u(e_i^1) = w_i + \epsilon_i$ for some $\epsilon_i \geq 0$. Note that e_i^0 was arbitrarily chosen. Hence, we can conclude for every $e_i^0 \in E^0$ with $\gamma_u(e_i^0) = w_i > 0$ and with complementary edge $e_i^1 \in E^1$ that $\gamma_u(e_i^1) = w_i + \epsilon_i$ for some $\epsilon_i \geq 0$. Hence,

$\sum_{e^1 \in E^1} \gamma_u(e^1) = \sum_{e^0 \in E^0} \gamma_u(e^0) + \sum_i \epsilon_i$. Now recall that both $\sum_{e^0 \in E^0} \gamma_u(e^0) = 1$ and $\sum_{e^1 \in E^1} \gamma_u(e^1) = 1$ hold. This is only possible if $\epsilon_i = 0$ for every i. In other words, $\gamma_u(e^0_i) = \gamma_u(e^1_i) = w_i$ for every $e^0_i \in E^0$ and its complementary edge $e^1_i \in E^1$. □

Lemma 3.6. *Let $\mathcal{F} = \langle T, (B_u)_{u \in T}, (\gamma_u)_{u \in T} \rangle$ be an FHD of width ≤2 of the hypergraph H constructed above and let $p \in [2n + 3; m]^-$. For every node u with $S \cup A'_p \cup \overline{A_p} \cup \{z_1, z_2\} \subseteq B_u$, the only way to cover $S \cup A'_p \cup \overline{A_p} \cup \{z_1, z_2\}$ by a fractional edge cover γ of weight ≤2 is by putting non-zero weight exclusively on edges $e^{k,0}_p$ and $e^{k,1}_p$ with $k \in \{1, 2, 3\}$. Moreover, $\sum_{k=1}^3 \gamma(e^{k,0}_p) = 1$ and $\sum_{k=1}^3 \gamma(e^{k,1}_p) = 1$ must hold.*

Proof. As in the proof of Lemma 3.5, to cover z_1, we have to put weight 1 on the edges in E^0 and to cover z_2, we have to put weight 1 on the edges in E^1, where E^0 and E^1 are defined as in the proof of Lemma 3.5. Since we have $width(\mathcal{F}) \leq 2$, we have to cover $A'_p \cup \overline{A_p} \cup S$ with the weight already put on the edges in $E^0 \cup E^1$. To cover A'_p, we have to put weight 1 on the edges in E^1_p, where

$$E^1_p = \{e^{k,1}_r \mid r \geq p\} \cup \{e^1_{\max}\}.$$

Notice that $E^1_p \subseteq E^1$ and therefore $\sum_{e \in E^1 \setminus E^1_p} \gamma_u(e) = 0$. Similarly, to cover $\overline{A_p}$, we have to put weight 1 on the edges in E^0_p, where

$$E^0_p = \{e^{k,0}_s \mid s \leq p\} \cup \{e^0_{(0,0)}\}.$$

Again, since $E^0_p \subseteq E^0$, $\sum_{e \in E^0 \setminus E^0_p} \gamma_u(e) = 0$. It remains to cover $S \cup \{z_1, z_2\}$. By Lemma 3.5, to cover S, z_1 and z_2, we have to put the same weight w on complementary edges e and e'. The only complementary edges in the sets E^0_p and E^1_p are edges of the form $e^{k,0}_p$ and $e^{k,1}_p$ with $k \in \{1, 2, 3\}$. In total, we thus have $\sum_{k=1}^3 e^{k,0}_p = 1$ and $\sum_{k=1}^3 e^{k,1}_p = 1$. □

Proof of the "only if"-direction. It remains to show that φ is satisfiable if H has a GHD or FHD of width ≤2. Due to the inequality $fhw(H) \leq ghw(H)$, it suffices to show that φ is satisfiable if H has an FHD of width ≤2. For this, let $\mathcal{F} = \langle T, (B_u)_{u \in T}, (\gamma_u)_{u \in T} \rangle$ be such an FHD. Let u_A, u_B, u_C and u'_A, u'_B, u'_C be the nodes that are guaranteed by Lemma 3.1. We state several properties of the path connecting u_A and u'_A, which heavily rely on Lemmas 3.5 and 3.6.

Claim A. *The nodes u'_A, u'_B, u'_C (respectively, u_A, u_B, u_C) are not on the path from u_A to u_C (respectively, u'_A to u'_C).*

Proof of Claim A. We only show that none of the nodes u'_i with $i \in \{A, B, C\}$ is on the path from u_A to u_C. The other property is shown analogously. Suppose to the contrary that some u'_i is on the path from u_A to u_C. Since u_B is also on the path between u_A and u_C, we distinguish two cases:

- Case (1): u'_i is on the path between u_A and u_B; then $\{b_1, b_2\} \subseteq B_{u'_i}$. This contradicts the property shown in Lemma 3.1 that u'_i cannot cover any vertices outside H'_0.
- Case (2): u'_i is on the path between u_B and u_C; then $\{c_1, c_2\} \subseteq B_{u'_i}$, which again contradicts Lemma 3.1.

Hence, the paths from u_A to u_C and from u'_A to u'_C are indeed disjoint. ◇

Claim B. *The following equality holds: $nodes(A \cup A', \mathcal{F}) \cap \{u_A, u_B, u_C, u'_A, u'_B, u'_C\} = \emptyset$.*

Proof of Claim B. Suppose to the contrary that there is a u_X (the proof for u'_X is analogous) for some $X \in \{A, B, C\}$, s.t. $u_X \in nodes(A \cup A', \mathcal{F})$; then there is some $a \in (A \cup A')$, s.t. $a \in B_{u_X}$. This contradicts the property shown in Lemma 3.1 that u_X cannot cover any vertices outside H_0. ◇

We are now interested in the sequence of nodes $\hat{u}_i$ that cover the edges $e^0_{(0,0)}, e_{\min}, e_{\min\oplus 1}, \ldots, e_{\max\ominus 1}, e_{\max}$. Before we formulate Claim C, it is convenient to introduce the following notation: To be able to refer to the edges $e^0_{(0,0)}$, $e_{\min}$, $e_{\min\oplus 1}$, $\ldots$, $e_{\max\ominus 1}$, $e^1_{\max}$ in a uniform way, we use $e_{\min\ominus 1}$ as synonym of $e^0_{(0,0)}$ and $e_{\max}$ as synonym of $e^1_{\max}$. We can thus define the natural order $e_{\min\ominus 1} < e_{\min} < e_{\min\oplus 1} < \cdots < e_{\max\ominus 1} < e_{\max}$ on these edges.

Claim C. *The FHD $\mathcal{F}$ has a path containing nodes $\hat{u}_1, \ldots, \hat{u}_N$ for some N, such that the edges $e_{\min\ominus 1}, e_{\min}, e_{\min\oplus 1}, \ldots, e_{\max\ominus 1}, e_{\max}$ are covered in this order. More formally, there is a mapping $f : \{\min\ominus 1, \ldots, \max\} \rightarrow \{1, \ldots, N\}$, s.t.*

- *$\hat{u}_{f(p)}$ covers e_p and*
- *if $p < p'$ then $f(p) \leq f(p')$.*

By a path containing nodes $\hat{u}_1, \ldots, \hat{u}_N$, *we mean that $\hat{u}_1$ and $\hat{u}_N$ are nodes in $\mathcal{F}$, such that the nodes $\hat{u}_2, \ldots, \hat{u}_{N-1}$ lie (in this order) on the path from $\hat{u}_1$ to $\hat{u}_N$. Of course, the path from $\hat{u}_1$ to $\hat{u}_N$ may also contain further nodes, but we are not interested in whether they cover any of the edges e_p.*

Proof of Claim C. Suppose to the contrary that no such path exists. Let $p \geq \min$ be the maximal value such that there is a path containing nodes $\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_\ell$, which cover $e_{\min\ominus 1}, \ldots, e_p$ in this order. Clearly, there exists a node $\hat{u}$ that covers $e_{p\oplus 1} = A'_{p\oplus 1} \cup \overline{A_{p\oplus 1}}$. We distinguish four cases:

- Case (1): $\hat{u}_1$ is on the path from $\hat{u}$ to all other nodes $\hat{u}_i$, with $1 < i \leq \ell$. By the connectedness condition, $\hat{u}_1$ covers A'_p. Hence, in total $\hat{u}_1$ covers $A'_p \cup A$ with $A'_p = \{a'_{\min}, \ldots, a'_p\}$ and $A = \{a_{\min}, \ldots, a_{\max}\}$. Then $\hat{u}_1$ covers all edges $e_{\min\ominus 1}, \ldots, e_p$. Therefore, the path containing nodes $\hat{u}_1$ and $\hat{u}$ covers $e_{\min\ominus 1}, \ldots, e_{p\oplus 1}$ in this order, which contradicts the maximality of p.
- Case (2): $\hat{u} = \hat{u}_1$, hence, $\hat{u}_1$ covers $A'_{p\oplus 1} \cup A$ with $A'_{p\oplus 1} = \{a'_{\min}, \ldots, a'_{p\oplus 1}\}$ and $A = \{a_{\min}, \ldots, a_{\max}\}$. Then, $\hat{u}_1$ covers all $e_{\min\ominus 1}, \ldots, e_{p\oplus 1}$, which contradicts the maximality of p.
- Case (3): $\hat{u}$ is on the path from $\hat{u}_1$ to $\hat{u}_\ell$ and $\hat{u} \neq \hat{u}_1$. Hence, $\hat{u}$ is between two nodes $\hat{u}_i$ and $\hat{u}_{i+1}$ for some $1 \leq i < \ell$ or $\hat{u} = \hat{u}_{i+1}$ for some $1 \leq i < \ell - 1$. The following arguments hold for both cases: Now, there is some $q \leq p$, such that e_q is covered by $\hat{u}_{i+1}$ and $e_{q\ominus 1}$ is covered by $\hat{u}_i$. Therefore, $\hat{u}$ covers $\overline{A_q}$ either by the connectedness condition (if $\hat{u}$ is between $\hat{u}_i$ and $\hat{u}_{i+1}$) or simply because $\hat{u} = \hat{u}_{i+1}$. Hence, in total, $\hat{u}$ covers $A'_{p\oplus 1} \cup \overline{A_q}$ with $A'_{p\oplus 1} = \{a'_{\min}, \ldots, a'_{p\oplus 1}\}$ and $\overline{A_q} = \{a_q, a_{q\oplus 1}, \ldots, a_p, a_{p\oplus 1}, \ldots a_{\max}\}$. Then, $\hat{u}$ covers all edges $e_q, e_{q\oplus 1}, \ldots, e_{p\oplus 1}$. Therefore, the path containing nodes $\hat{u}_1, \ldots, \hat{u}_i, \hat{u}$ covers $e_{\min\ominus 1}, \ldots, e_{p\oplus 1}$ in this order, which contradicts the maximality of p.
- Case (4): There is a u^* on the path from $\hat{u}_1$ to $\hat{u}_\ell$, such that the paths from $\hat{u}_1$ to $\hat{u}$ and from $\hat{u}$ to $\hat{u}_\ell$ go through u^* and, moreover, $u^* \neq \hat{u}_1$. Then, u^* is either between $\hat{u}_i$ and $\hat{u}_{i+1}$ for some $1 \leq i < \ell$ or $u^* = \hat{u}_{i+1}$ for some $1 \leq i < \ell - 1$. The following arguments hold for both cases: There is some $q \leq p$, such that e_q is covered by $\hat{u}_{i+1}$ and $e_{q\ominus 1}$ is covered by $\hat{u}_i$. By the connectedness condition, u^* covers
  - $A'_p = \{a'_{\min}, \ldots, a'_p\}$, since u^* is on the path from $\hat{u}$ to $\hat{u}_\ell$, and
  - $\overline{A}_q = \{a_q, \ldots, a_p, a_{p\oplus 1}, \ldots a_{\max}\}$, since u^* is on the path from $\hat{u}_1$ to $\hat{u}_{i+1}$ or $u^* = \hat{u}_{i+1}$.

  Then u^* covers all edges $e_q, e_{q\oplus 1}, \ldots, e_p$. Therefore, the path containing the nodes $\hat{u}_1, \ldots, \hat{u}_i$, u^*, $\hat{u}$ covers $e_{\min\ominus 1}, \ldots, e_{p\oplus 1}$ in this order, which contradicts the maximality of p. ◇

So far, we have shown that there are three disjoint paths from u_A to u_C, from u'_A to u'_C and from $\hat{u}_1$ to $\hat{u}_N$, respectively. It is easy to see that u_A is closer to the path $\hat{u}_1, \ldots, \hat{u}_N$ than u_B and u_C, since otherwise u_B and u_C would have to cover a_1 as well, which is impossible by Lemma 3.1. The same also holds for u'_A. In the next claims, we will argue that the path from u_A to u'_A goes through some

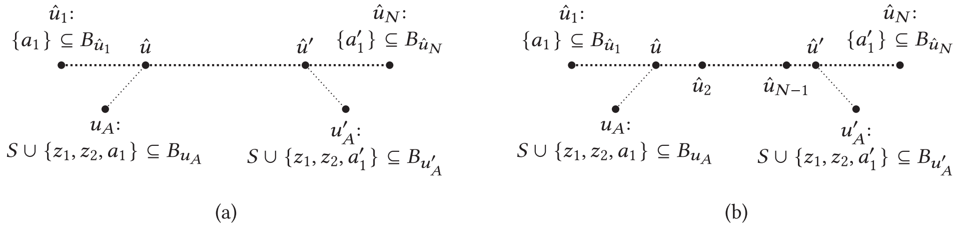
$\hat{u}_1$:
$\{a_1\} \subseteq B_{\hat{u}_1}$
$\hat{u}$
$\hat{u}'$
$\hat{u}_N$:
$\{a'_1\} \subseteq B_{\hat{u}_N}$
u_A:
$S \cup \{z_1, z_2, a_1\} \subseteq B_{u_A}$
u'_A:
$S \cup \{z_1, z_2, a'_1\} \subseteq B_{u'_A}$
(a)
$\hat{u}_1$:
$\{a_1\} \subseteq B_{\hat{u}_1}$
$\hat{u}$
$\hat{u}_2$
$\hat{u}_{N-1}$
$\hat{u}'$
$\hat{u}_N$:
$\{a'_1\} \subseteq B_{\hat{u}_N}$
u_A:
$S \cup \{z_1, z_2, a_1\} \subseteq B_{u_A}$
u'_A:
$S \cup \{z_1, z_2, a'_1\} \subseteq B_{u'_A}$
(b)

Fig. 4. Arrangement of the nodes $\hat{u}_1$, $\hat{u}$, $\hat{u}'$, and $\hat{u}_N$ from Claim E (a) and Claim G (b).

node $\hat{u}$ of the path from $\hat{u}_1$ to $\hat{u}_N$. We write $\pi(\hat{u}_1, \hat{u}_N)$ as a short-hand notation for the path from $\hat{u}_1$ to $\hat{u}_N$. Next, we state some important properties of $\pi(\hat{u}_1, \hat{u}_N)$ and the path from u_A to u'_A.

Claim D. *In the FHD $\mathcal{F}$ of H of width ≤ 2, the path from u_A to u'_A has non-empty intersection with $\pi(\hat{u}_1, \hat{u}_N)$.*

Proof of Claim D. Suppose to the contrary that the path from u_A to u'_A is disjoint from $\pi(\hat{u}_1, \hat{u}_N)$. We distinguish three cases:

- Case (1): u_A is on the path from u'_A to (some node in) $\pi(\hat{u}_1, \hat{u}_N)$. Then, by the connectedness condition, u_A must contain a'_1, which contradicts Lemma 3.1.
- Case (2): u'_A is on the path from u_A to $\pi(\hat{u}_1, \hat{u}_N)$. Analogously to Case (1), we get a contradiction by the fact that then u'_A must contain a_1.
- Case (3): There is a node u^* on the path from u_A to u'_A, which is closest to $\pi(\hat{u}_1, \hat{u}_N)$, i.e., u^* lies on the path from u_A to u'_A and both paths, the one connecting u_A with $\pi(\hat{u}_1, \hat{u}_N)$ and the one connecting u'_A with $\pi(\hat{u}_1, \hat{u}_N)$, go through u^*. Hence, by the connectedness condition, the bag of u^* contains $S \cup \{z_1, z_2, a_1, a'_1\}$. By Lemma 3.5, to cover $S \cup \{z_1, z_2\}$ with weight ≤ 2, we are only allowed to put non-zero weight on pairs of complementary edges. However, then it is impossible to achieve also weight ≥ 1 on a_1 and a'_1 at the same time. ◇

Claim E. *In the FHD $\mathcal{F}$ of H of width ≤ 2 there are two distinct nodes $\hat{u}$ and $\hat{u}'$ in the intersection of the path from u_A to u'_A with $\pi(\hat{u}_1, \hat{u}_N)$, s.t. $\hat{u}$ is the node in $\pi(\hat{u}_1, \hat{u}_N)$ closest to u_A and $\hat{u}'$ is the node in $\pi(\hat{u}_1, \hat{u}_N)$ closest to u'_A. Then, on the path $\pi(\hat{u}_1, \hat{u}_N)$, $\hat{u}$ comes before $\hat{u}'$. See Figure 4 (a) for a graphical illustration of the arrangement of the nodes $\hat{u}_1$, $\hat{u}$, $\hat{u}'$, and $\hat{u}_N$ on the path $\pi(\hat{u}_1, \hat{u}_N)$.*

Proof of Claim E. First, we show that $\hat{u}$ and $\hat{u}'$ are indeed distinct. Suppose towards a contradiction that they are not, i.e., $\hat{u} = \hat{u}'$. Then, by connectedness, $\hat{u}$ has to cover $S \cup \{z_1, z_2\}$, because $S \cup \{z_1, z_2\}$ is contained in B_{u_A} and in $B_{u'_A}$. Moreover, again by connectedness, $\hat{u}$ also has to cover $\{a_1, a'_1\}$, because a_1 is contained in $B_{\hat{u}_1}$ and in B_{u_A} and a'_1 is contained in $B_{\hat{u}_N}$ and in $B_{u'_A}$. As in Case (3) in the proof of Claim D, this is impossible by Lemma 3.5. Hence, $\hat{u}$ and $\hat{u}'$ are distinct.

Second, we show that, on the path from $\hat{u}_1$ to $\hat{u}_N$, the node $\hat{u}$ comes before $\hat{u}'$. Suppose to the contrary that $\hat{u}'$ comes before $\hat{u}$. Then, by the connectedness condition, $\hat{u}$ covers the following (sets of) vertices:

- a'_1, since we are assuming that $\hat{u}'$ comes before $\hat{u}$, i.e., $\hat{u}$ is on the path from $\hat{u}_N$ to u'_A;
- a_1, since $\hat{u}$ is on the path from $\hat{u}_1$ to u_A;
- $S \cup \{z_1, z_2\}$, since $\hat{u}$ is on the path from u_A to u'_A.

In total, $\hat{u}$ has to cover all vertices in $S \cup \{z_1, z_2, a_1, a'_1\}$. Again, by Lemma 3.5, this is impossible with weight ≤ 2. ◇

Claim F. *In the FHD $\mathcal{F}$ of H of width ≤ 2 the path $\pi(\hat{u}_1, \hat{u}_N)$ has at least three nodes $\hat{u}_i$, i.e., $N \geq 3$.*

Proof of Claim F. First, it is easy to verify that $N \geq 2$ must hold. Otherwise, a single node would have to cover $\{e_{\min \ominus 1}, e_{\min}, e_{\min \oplus 1}, \dots, e_{\max \ominus 1}, e_{\max}\}$ and, hence, in particular, $S \cup \{z_1, z_2, a_1, a'_1\}$, which is impossible, as we have already seen in Case (3) of the proof of Claim D.

It remains to prove $N \geq 3$. Suppose to the contrary that $N = 2$. By the problem reduction, hypergraph H has distinct edges $e_{\min \ominus 1}$, $e_{\min}$ and $e_{\max}$. Hence, $\hat{u}_1$ covers at least $e_{\min \ominus 1}$ and $\hat{u}_2$ covers at least $e_{\max}$. Recall from Claim E the nodes $\hat{u}$ and $\hat{u}'$, which constitute the endpoints of the intersection of the path from u_A to u'_A with the path $\pi(\hat{u}_1, \hat{u}_N)$ (cf. Figure 4(a)). Here, we are assuming $N = 2$. We now show that, by the connectedness condition of FHDs, the nodes $\hat{u}$ and $\hat{u}'$ must cover certain vertices, which will lead to a contradiction by Lemma 3.5.

- vertices covered by $\hat{u}$: node $\hat{u}$ is on the path between u_A and u'_A. Hence, it covers $S \cup \{z_1, z_2\}$. Moreover, $\hat{u}$ is on the path between $\hat{u}_1$ and u_A (or even coincides with $\hat{u}_1$). Hence, it also covers a_1. In total, $\hat{u}$ covers at least $S \cup \{z_1, z_2, a_1\}$.
- vertices covered by $\hat{u}'$: node $\hat{u}'$ is on the path between u_A and u'_A. Hence, it covers $S \cup \{z_1, z_2\}$. Moreover, $\hat{u}'$ is on the path between $\hat{u}_2$ and u'_A (or even coincides with $\hat{u}_2$). Hence, it also covers a'_1. In total, $\hat{u}'$ covers at least $S \cup \{z_1, z_2, a'_1\}$.

One of the nodes $\hat{u}_1$ or $\hat{u}_2$ must also cover the edge $e_{\min}$. We inspect these two cases separately:

- Case (1): suppose that the edge $e_{\min}$ is covered by $\hat{u}_1$. Then, $\hat{u}_1$ covers vertex $a'_{\min}$, which is also covered by $\hat{u}_2$. Hence, also $\hat{u}$ covers $a'_{\min}$. In total, $\hat{u}$ covers $S \cup \{z_1, z_2, a_1, a'_{\min}\}$. However, by Lemma 3.5, we know that, to cover $S \cup \{z_1, z_2\}$ with weight ≤ 2, we are only allowed to put non-zero weight on pairs of complementary edges. Hence, it is impossible to achieve also weight ≥ 1 on a_1 and on $a'_{\min}$ at the same time.
- Case (2): suppose that the edge $e_{\min}$ is covered by $\hat{u}_2$. Then, $\hat{u}_2$ covers vertex $a_{\min}$ (actually, it even covers all of A), which is also covered by $\hat{u}_1$. Hence, also $\hat{u}'$ covers $a_{\min}$. In total, $\hat{u}'$ covers $S \cup \{z_1, z_2, a'_1, a_{\min}\}$. Again, this is impossible by Lemma 3.5.

Hence, the path $\pi(\hat{u}_1, \hat{u}_N)$ indeed has at least three nodes $\hat{u}_i$. ◇

Claim G. *In the FHD $\mathcal{F}$ of H of width ≤ 2 all the nodes $\hat{u}_2, \dots, \hat{u}_{N-1}$ are on the path from u_A to u'_A. For the nodes $\hat{u}$ and $\hat{u}'$ from Claim E, this means that the nodes $\hat{u}_1, \hat{u}, \hat{u}_2, \hat{u}_{N-1}, \hat{u}', \hat{u}_N$ are arranged in precisely this order on the path $\pi(\hat{u}_1, \hat{u}_N)$ from $\hat{u}_1$ to $\hat{u}_N$ (cf. Figure 4(b)). The node $\hat{u}$ may possibly coincide with $\hat{u}_1$ and $\hat{u}'$ may possibly coincide with $\hat{u}_N$.*

Proof of Claim G. We have to prove that $\hat{u}$ lies between $\hat{u}_1$ and $\hat{u}_2$ (not including $\hat{u}_2$) and $\hat{u}'$ lies between $\hat{u}_{N-1}$ and $\hat{u}_N$ (not including $\hat{u}_{N-1}$). For the first property, suppose to the contrary that $\hat{u}$ does not lie between $\hat{u}_1$ and $\hat{u}_2$ or $\hat{u} = \hat{u}_2$. This means that there exists $i \in \{2, \dots, N-1\}$ such that $\hat{u}$ lies between $\hat{u}_i$ and $\hat{u}_{i+1}$, including the case that $\hat{u}$ coincides with $\hat{u}_i$. Note that, by Claim E, $\hat{u}$ cannot coincide with $\hat{u}_N$, since there is yet another node $\hat{u}'$ between $\hat{u}$ and $\hat{u}_N$.

By definition of $\hat{u}_i$ and $\hat{u}_{i+1}$, there is a $p \in [2n+3; m]$, such that both $\hat{u}_i$ and $\hat{u}_{i+1}$ cover a'_p. Then, by the connectedness condition, $\hat{u}$ covers the following (sets of) vertices:

- a'_p, since $\hat{u}$ is on the path from $\hat{u}_i$ to $\hat{u}_{i+1}$ (or $\hat{u}$ coincides with $\hat{u}_i$),
- a_1, since $\hat{u}$ is on the path from $\hat{u}_1$ to u_A,
- $S \cup \{z_1, z_2\}$, since $\hat{u}$ is on the path from u_A to u'_A.

However, by Lemma 3.5, we know that, to cover $S \cup \{z_1, z_2\}$ with weight ≤ 2, we are only allowed to put non-zero weight on pairs of complementary edges. Hence, it is impossible to achieve also weight ≥ 1 on a'_p and a_1 at the same time.

It remains to show that $\hat{u}'$ lies between $\hat{u}_{N-1}$ and $\hat{u}_N$ (not including $\hat{u}_{N-1}$). Suppose to the contrary that it does not. Then, analogously to the above considerations for $\hat{u}$, it can be shown that there exists some $p \in [2n+3; m]$, such that $\hat{u}'$ covers the vertices $S \cup \{z_1, z_2, a_p, a'_1\}$. Again, this is impossible by Lemma 3.5. ◇

By Claim C, the decomposition $\mathcal{F}$ contains a path $\hat{u}_1 \cdots \hat{u}_N$ that covers the edges $e_{\min \ominus 1}, e_{\min}, e_{\min \oplus 1}, \dots, e_{\max \ominus 1}, e_{\max}$ in this order. We next strengthen this property by showing that every node $\hat{u}_i$ covers exactly one edge e_p.

Claim H. *Each of the nodes $\hat{u}_1, \dots, \hat{u}_N$ covers exactly one of the edges $e_{\min \ominus 1}$, $e_{\min}$, $e_{\min \oplus 1}$, $\dots$, $e_{\max \ominus 1}$, $e_{\max}$.*

Proof of Claim H. We prove this property for the "outer nodes" $\hat{u}_1, \hat{u}_N$ and for the "inner nodes" $\hat{u}_2 \cdots \hat{u}_{N-1}$ separately. We start with the "outer nodes." The proof for $\hat{u}_1$ and $\hat{u}_N$ is symmetric. We thus only work out the details for $\hat{u}_1$. Suppose to the contrary that $\hat{u}_1$ not only covers $e_{\min \ominus 1}$ but also $e_{\min}$. We distinguish two cases according to the position of node $\hat{u}$ in Figure 4(b):

- Case (1): $\hat{u} = \hat{u}_1$. Then, $\hat{u}_1$ has to cover the following (sets of) vertices:
  — $S \cup \{z_1, z_2\}$, since $\hat{u}$ is on the path from u_A to u'_A and we are assuming $\hat{u} = \hat{u}_1$.
  — a_1, since $\hat{u}_1$ covers $e_{\min \ominus 1}$,
  — $a'_{\min}$, since we are assuming that $\hat{u}_1$ also covers $e_{\min}$.
  By applying Lemma 3.5, we may conclude that the set $S \cup \{z_1, z_2, a_1, a'_{\min}\}$ cannot be covered by a fractional edge cover of weight ≤2.
- Case (2): $\hat{u} \neq \hat{u}_1$. Then $\hat{u}$ is on the path from $\hat{u}_1$ to $\hat{u}_2$. Hence, $\hat{u}$ has to cover the following (sets of) vertices:
  — $S \cup \{z_1, z_2\}$, since $\hat{u}$ is on the path from u_A to u'_A,
  — a_1, since $\hat{u}$ is on the path from u_A to $\hat{u}_1$,
  — $a'_{\min}$, since $\hat{u}$ is on the path from $\hat{u}_1$ to $\hat{u}_2$.
  As in Case (1) above, $S \cup \{z_1, z_2, a_1, a'_{\min}\}$ cannot be covered by a fractional edge cover of weight ≤2 due to Lemma 3.5.

It remains to consider the "inner" nodes $\hat{u}_i$ with $2 \leq i \leq N-1$. Each such $\hat{u}_i$ has to cover $S \cup \{z_1, z_2\}$, since all these nodes are on the path from u_A to u'_A by Claim G. Now suppose that $\hat{u}_i$ covers $e_p = A'_p \cup \overline{A_p}$ for some $p \in \{e_{\min}, \dots, e_{\max \ominus 1}\}$. By Lemma 3.6, covering all of the vertices $A'_p \cup \overline{A_p} \cup S \cup \{z_1, z_2\}$ by a fractional edge cover of weight ≤2 requires that we put total weight 1 on the edges $e_p^{k,0}$ and total weight 1 on the edges $e_p^{k,1}$ with $k \in \{1,2,3\}$. However, then it is impossible to cover also $e_{p'}$ for some p' with $p' \neq p$. This concludes the proof of Claim F. ◇

We can now associate with each $\hat{u}_i$ for $1 \leq i \leq N$ the corresponding edge e_p and write u_p to denote the node that covers the edge e_p. By Claim G, we know that all of the nodes $u_{\min} \dots, u_{\max \ominus 1}$ are on the path from u_A to u'_A. Hence, by the connectedness condition, all these nodes cover $S \cup \{z_1, z_2\}$.

We are now ready to construct a satisfying truth assignment σ of φ. For each $i \leq 2n+3$, let X_i be the set $B_{u_{(i,1)}} \cap (Y \cup Y')$. As $Y \subseteq B_{u_A}$ and $Y' \subseteq B_{u'_A}$, the sequence $X_1 \cap Y, \dots, X_{2n+3} \cap Y$ is non-increasing and the sequence $X_1 \cap Y', \dots, X_{2n+3} \cap Y'$ is non-decreasing. Furthermore, as all edges $e_{y_i} = \{y_i, y'_i\}$ must be covered by some node in $\mathcal{F}$, we conclude that for each i and j, $y_j \in X_i$ or $y'_j \in X_i$. Then, there is some $s \leq 2n+2$ such that $X_s = X_{s+1}$. Furthermore, all nodes between $u_{(s,1)}$ and $u_{(s+1,1)}$ cover X_s. We derive a truth assignment for $x_1, \dots, x_n$ from X_s as follows: For each $\ell \leq n$, we set $\sigma(x_\ell) = 1$ if $y_\ell \in X_s$ and otherwise $\sigma(x_\ell) = 0$. Note that in the latter case $y'_\ell \in X_s$.

Claim I. *The constructed truth assignment σ is a model of φ.*

Proof of Claim I. We have to show that every clause $c_j = L_j^1 \vee L_j^2 \vee L_j^3$ of φ is true in σ. Choose an arbitrary $j \in \{1, \ldots, m\}$. We have to show that there exists a literal in c_j that is true in σ. To this end, we inspect the node $u_{(s,j)}$, which, by construction, lies between $u_{(s,1)}$ and $u_{(s+1,1)}$. Let $p = (s, j)$. Then, we have $A'_p \cup \overline{A_p} \cup S \cup \{z_1, z_2\} \subseteq B_{u_p}$. Moreover, by the definition of X_s, we also have $X_s \subseteq B_{u_p}$. By Lemma 3.6, the only way to cover B_{u_p} with weight ≤ 2 is by using exclusively the edges $e_p^{k,0}$ and $e_p^{k,1}$ with $k \in \{1, 2, 3\}$. More specifically, we have $\sum_{k=1}^{3} \gamma_{u_p}(e_p^{k,0}) = 1$ and $\sum_{k=1}^{3} \gamma_{u_p}(e_p^{k,1}) = 1$. Therefore, $\gamma_{u_p}(e_p^{k,0}) > 0$ for some k. We distinguish two cases depending on the form of literal L_j^k:

- Case (1): First, suppose $L_j^k = x_\ell$. By Lemma 3.5, complementary edges must have equal weight. Hence, from $\gamma_{u_p}(e_p^{k,0}) > 0$ it follows that also $\gamma_{u_p}(e_p^{k,1}) > 0$ holds. Thus, the weight on y'_ℓ is less than 1, which means that $y'_\ell \notin B(\gamma_{u_p})$ and consequently $y'_\ell \notin X_s$. Since this implies that $y_\ell \in X_s$, we indeed have that $\sigma(x_\ell) = 1$.
- Case (2): Conversely, suppose $L_j^k = \neg x_\ell$. Since $\gamma_{u_p}(e_p^{k,0}) > 0$, the weight on y_ℓ is less than 1, which means that $y_\ell \notin B(\gamma_{u_p})$ and consequently $y_\ell \notin X_s$. Hence, we have $\sigma(x_\ell) = 0$.

In either case, literal L_p^k is satisfied by σ and therefore, the jth clause c_j is satisfied by σ. Since j was arbitrarily chosen, σ indeed satisfies φ. ◇

Claim I completes the proof of Theorem 3.2. □

We conclude this section by mentioning that the above reduction is easily extended to $k + \ell$ for arbitrary $\ell \geq 1$: For integer values ℓ, simply add a clique of 2ℓ fresh vertices $v_1, \ldots, v_{2\ell}$ to H and connect each v_i with each "old" vertex in H. Now assume a rational value $\ell \geq 1$, i.e., $\ell = r/q$ for natural numbers r, q with $r > q > 0$. To achieve a rational bound $k + r/q$, we add r fresh vertices and add hyperedges $\{v_i, v_{i\oplus 1}, \ldots, v_{i\oplus(q-1)}\}$ with $i \in \{1, \ldots, r\}$ to H, where $a \oplus b$ denotes $a + b$ modulo r. Again, we connect each v_i with each "old" vertex in H. With this construction, we can give NP-hardness proofs for any (fractional) $k \geq 3$. For all fractional values $k < 3$ (except for $k = 2$) different gadgets and ideas might be needed to prove NP-hardness of Check(FHD,k), which we leave for future work.

4 A FRAMEWORK FOR EFFICIENT COMPUTATION OF DECOMPOSITIONS

Before we move on to the easy cases for Check(GHD, k) and Check(FHD, k), we will introduce a framework for a uniform presentation of the tractability proofs in the following sections. Conceptually, we can split the task of checking whether a decomposition of certain width exists into two parts: (1) deciding which sets of vertices are acceptable as bags (the *candidate bags*) and (2) deciding if there is a tree decomposition made up of only acceptable bags. In this section, we focus on the second part and show that this task is indeed tractable as long as the decompositions satisfy a certain normal form. This will allow us to show the Check problem tractable for settings where we can compute an appropriate set of candidate bags in polynomial time.

To emphasize the generality of the approach, we will focus on tree decompositions in this section. Recall that a generalized hypertree decomposition of width at most k is simply a tree decomposition where every bag has an integral edge cover with weight at most k. The same is true for fractional hypertree decompositions and fractional edge covers. Hence, the Check(GHD, k) problem can be solved by computing appropriate sets $\mathbf{S}$ of candidate bags that can be covered by k edges and then deciding whether there exists a TD using only bags from $\mathbf{S}$. If such a TD exists, then it is a witness for the existence of a GHD of width at most k. Of course, the same strategy also works for Check(FHD, k).

First, we will formally define the task we are interested in as the candidate tree decomposition problem. We show that the problem is NP-complete even for acyclic graphs. Following that, we show that the problem becomes tractable if we limit our search to finding TDs that adhere to a certain normal form that is sufficient for our purposes.

Definition 4.1. Let H be a hypergraph and $\mathcal{T} = \langle T, (B_u)_{u \in T} \rangle$ be a tree decomposition of H. Let the *candidate bags* $\mathbf{S}$ be a family of subsets of $V(H)$. If for each $u \in T$ there exists an $S \in \mathbf{S}$ such that $B_u = S$, then we call $\mathcal{T}$ a *candidate tree decomposition* of $\mathbf{S}$. We denote by $\mathbf{CTD}(\mathbf{S})$ the set of all candidate tree decompositions of H.

Theorem 4.2. *Let H be a hypergraph and $\mathbf{S} \subseteq 2^{V(H)}$. It is* NP*-complete to decide whether* $\mathbf{CTD}(\mathbf{S}) \neq \emptyset$*. The problem remains* NP*-complete even if we restrict the choice of H to acyclic graphs.*

Proof. The problem is clearly in NP. We show NP-hardness by reduction from the exact cover problem: Let $U = \{u_1, \dots, u_n\}$ be the universe and let $X_1, \dots, X_m$ be subsets of U. The exact cover problem asks for a cover of U by elements of $\{X_1, \dots, X_m\}$ such that the sets in the cover are pairwise disjoint.

We define an acyclic graph G as follows: G is a tree with vertices $v, v_1, \dots .v_n, u_1, \dots, u_n$. The edges of G are $\{v, v_i\}$ and $\{v_i, u_i\}$ for each $1 \leq i \leq n$. Let $\mathbf{S} = \{S, S_1, \dots, S_m\}$ where $S = \{v, v_1, \dots v_n\}$ and each $S_i = X_i \cup \{v_j \mid u_j \in X_i\}$, i.e., by taking X_i and adding v_j for each u_j contained in X_i.

We claim that then $\mathbf{CTD}(\mathbf{S}) \neq \emptyset$ iff there is an exact cover of U by $X_1, \dots, X_m$. One direction is easy. Let $\mathbf{X} \subseteq \{X_1, \dots, X_m\}$ be an exact cover of U. Denote the elements of $\mathbf{X}$ by $X'_1, \dots, X'_q$. For $1 \leq i \leq q$, let S'_i be obtained from X'_i by adding v_i for each $u_i \in X'_i$. Clearly $S'_i \in \{S_1, \dots, S_m\}$ and there exists a candidate tree decomposition $\langle T, (B_u)_{u \in T} \rangle$ of G where $V(T) = \{t, t_1, \dots, t_q\}$, $E(T) = \{\{t, t_1\}, \dots, \{t, t_q\}\}$, $B_t = S$ and for each $1 \leq i \leq q$, $B_{t_i} = S'_i$.

The other direction is more complicated. Let $\langle T, (B_u)_{u \in T} \rangle$ be a smallest (w.r.t. the number of nodes of T) candidate TD of G. In particular, this means that the bags of all the nodes are distinct. The proof proceeds in several steps:

(1) There is $t \in V(T)$ such that $B_t = S$. Indeed, otherwise, v is not covered.

(2) For each $1 \leq i \leq n$, there is $t' \in V(T)$ such that $u_i \in B_{t'}$ and t is adjacent to t'. Indeed, assume the opposite and let $t' \in V(T)$ be a node with $u_i \in B_{t'}$ such that t is not adjacent to t'.

Since we assume a candidate TD, we have $B_{t'} = S_j$ for some j and hence $v_i \in B_{t'}$. Now consider the path between t and t' and let s be the node next to t on this path. By our minimality assumption, we know that $B_s \neq B_t$ and thus $B_s = S_k$ with $k \neq j$. Since we assume the claim to be false, we have $u_i \notin S_k$ and hence $v_i \notin S_k$. As $v_i \in B_{t'}$ and $v_i \in B_t$, the connectedness condition of the tree decomposition is violated.

(3) Let $t_1, \dots, t_q$ be the neighbors of t in T. We claim that T has no other nodes. Indeed, all the vertices of G are covered by the bags of $t, t_1, \dots t_q$ by the previous two items. Each edge $\{v, v_i\}$ is contained in B_t. Also, each edge $\{u_i, v_i\}$ is covered by some B_{t_j} containing u_i (existing by the previous item). It follows that $T[\{t, t_1, \dots, t_q\}]$ together with the corresponding bags form a tree decomposition of G. By the minimality assumption, T does not have other nodes.

(4) We claim that for any $1 \leq i \neq j \leq q$, $B_{t_i} \cap B_{t_j} \neq \emptyset$. Indeed, otherwise, there is $u_k \in B_{t_i} \cap B_{t_j}$. However, $u_k \notin B_t$ in contradiction to the connectedness condition. It follows that $B_{t_1} \cap U, \dots, B_{t_q} \cap U$ are disjoint elements of $\{X_1, \dots, X_m\}$ covering all of U as required. □

To obtain a tractable version of the problem, we will introduce a generalization of the normal form that was used in the tractability proof for HDs in Reference [31].

Definition 4.3. A tree decomposition $\langle T, (B_u)_{u \in T} \rangle$ of a hypergraph H is in *component normal form (ComNF)* if for each node $r \in T$, and for each child s of r there is *exactly one* $[B_r]$-component C_s such that $V(T_s) = C_s \cup (B_r \cap B_s)$ holds. We say C_s is the component *associated with* node s.

By the above definition, we know that in in a ComNF TD, every child s of a node r is associated with at most one $[B_r]$-component C_s. By the connectedness condition, also the converse is true, i.e., every $[B_r]$-component C_s is associated with *at most one* child s. Note that (by applying the ideas of the transformation of HDs into the normal form of Reference [31]), every TD can be transformed in polynomial time into a TD in ComNF without increasing the width (more precisely, the bags in the resulting TD are subsets of the bags in the original TD).

Definition 4.4. Let H be a hypergraph and let $\mathbf{S}$ be a family of subsets of $V(H)$. Let $\mathcal{T} \in \mathbf{CTD}(\mathbf{S})$ be a tree decomposition in ComNF. We say $\mathcal{T}$ is a *ComNF candidate tree decomposition* of $\mathbf{S}$. We denote by $\mathbf{ComCTD}(\mathbf{S})$ the set of all ComNF candidate tree decompositions of H.

Theorem 4.5. *Let H be a hypergraph and $\mathbf{S} \subseteq 2^{V(H)}$. There exists an algorithm that takes H and $\mathbf{S}$ as an input and decides in polynomial time whether $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$, and if so, return a $\mathcal{T} \in \mathbf{ComCTD}(\mathbf{S})$.*

The intuition behind a polynomial-time algorithm for this problem is simple. Every parent/child relationship in a tree decomposition corresponds to a separator S (the bag of the parent) and an $[S]$-component. It should therefore be enough to first enumerate all pairs (S, C) of separators S in $\mathbf{S}$ and $[S]$-components C, and then check if these pairs, which we will call *blocks*, can be combined to form a valid tree decomposition. Through the restriction to a specific set $\mathbf{S}$, that is part of the input, the number of blocks we have to consider is only polynomial in the input.

We are not aware of a proof of Theorem 4.5 in the literature. In light of the hardness result for the general case, we choose to present a full proof of the theorem in Appendix A even though it could be considered folklore. In particular, methods for subedge-based decompositions [32], as well as tree projections and their associated algorithms (see e.g., References [27, 29, 43]), are closely related to the ComNF CTD problem.

The restriction to component normal form will ultimately not restrict us in the following sections. For generalized and fractional hypertree width, ComNF can be enforced without increasing the width. Such a transformation can be found, e.g, in Reference [31] and as part of the proof of Lemma 5.9. Still, some care will be required in the enumeration of the candidate bags to guarantee that they allow for a decomposition in component normal form.

5 TRACTABLE CASES OF GHD COMPUTATION

As discussed in Section 1, we are interested in finding a realistic and non-trivial criterion on hypergraphs that makes the Check(GHD, k) problem tractable for fixed k. We thus propose here such a simple property, namely, the bounded intersection of two or more edges.

Definition 5.1. The *intersection width iwidth*(H) of a hypergraph H is the maximum cardinality of any intersection $e_1 \cap e_2$ of two distinct edges e_1 and e_2 of H. We say that a hypergraph H has the ***i-bounded intersection property (i-BIP)*** if $iwidth(H) \leq i$ holds.

Let $\mathscr{C}$ be a class of hypergraphs. We say that $\mathscr{C}$ has the ***bounded intersection property (BIP)*** if there exists some integer constant i such that every hypergraph H in $\mathscr{C}$ has the i-BIP. Class $\mathscr{C}$ has the ***logarithmically-bounded intersection property (LogBIP)*** if for each of its elements H, $iwidth(H)$ is $O(\log n)$, where $n = ||H||$ denotes the size of H.

The BIP criterion properly generalizes bounded arity and is indeed non-trivial in the sense that there exist classes of unbounded *ghw* that enjoy the BIP. Among others this includes the classes

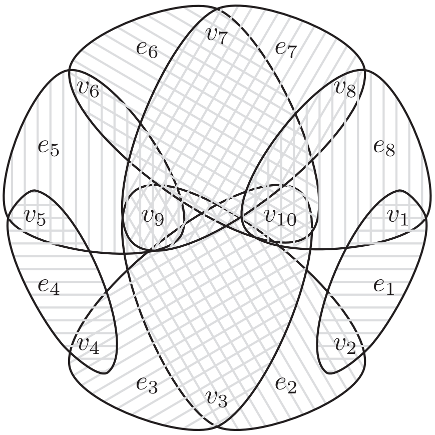

Fig. 5. Hypergraph H_0 from Example 5.3.

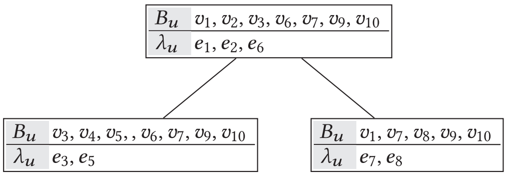
B_u $v_1, v_2, v_3, v_6, v_7, v_9, v_{10}$
λ_u e_1, e_2, e_6
B_u $v_3, v_4, v_5, , v_6, v_7, v_9, v_{10}$
λ_u e_3, e_5
B_u $v_1, v_7, v_8, v_9, v_{10}$
λ_u e_7, e_8

Fig. 6. HD of hypergraph H_0 in Figure 5.

of graphs, regular hypergraphs, and linear hypergraphs. Moreover, a recent empirical study [23] suggests that the overwhelming number of CQs enjoys the 2-BIP (i.e., one hardly joins two relations over more than two attributes). To allow for a yet bigger class of hypergraphs, the BIP can be relaxed as follows:

Definition 5.2. The *c-multi-intersection width c-miwidth(H)* of a hypergraph H is the maximum cardinality of any intersection $e_1 \cap \cdots \cap e_c$ of c distinct edges $e_1, \ldots, e_c$ of H. We say that a hypergraph H has the ***i-bounded c-multi-intersection property (ic-BMIP)*** if c-*miwidth(H)* $\leq i$ holds.

Let $\mathscr{C}$ be a class of hypergraphs. We say that $\mathscr{C}$ has the ***bounded multi-intersection property (BMIP)*** if there exist constants c and i such that every hypergraph H in $\mathscr{C}$ has the ic-BMIP. Class $\mathscr{C}$ of hypergraphs has the ***logarithmically-bounded multi-intersection property (LogBMIP)*** if there is a constant c such that for the hypergraphs $H \in \mathscr{C}$, c-*miwidth(H)* is $O(\log n)$, where n denotes the size of the hypergraph H.

Example 5.3. Figure 5 shows the hypergraph $H_0 = (V_0, E_0)$ with $ghw(H_0) = 2$ but $hw(H_0) = 3$. (This example is from Reference [32], which, in turn, was inspired by work of Adler [3].) Figure 6 shows an HD of width 3, and Figure 7 shows GHDs of width 2 for the hypergraph H_0. The *iwidth* and the 3-*miwidth* of H_0 is 1. Starting from $c = 4$, the c-*miwidth* is 0. ◇

The LogBMIP is the most liberal restriction on classes of hypergraphs introduced in Definitions 5.1 and 5.2. The main result in this section will be that the Check(GHD, k) problem with fixed k is tractable for any class of hypergraphs satisfying this criterion.

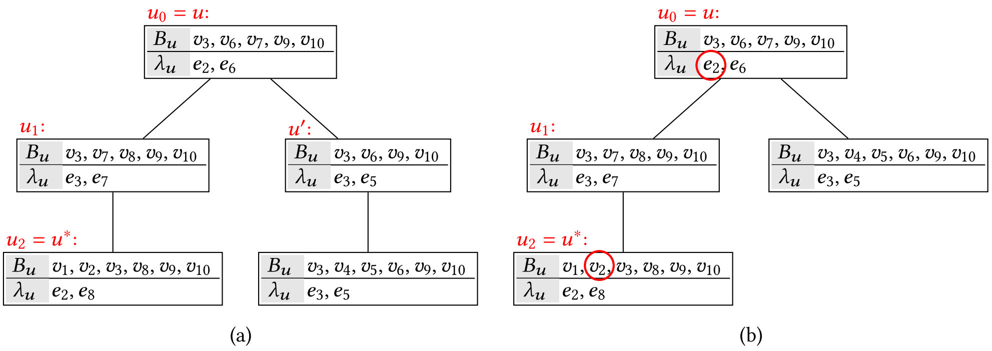
$u_0 = u$:
B_u $v_3, v_6, v_7, v_9, v_{10}$
λ_u e_2, e_6
u_1:
B_u $v_3, v_7, v_8, v_9, v_{10}$
λ_u e_3, e_7
u':
B_u v_3, v_6, v_9, v_{10}
λ_u e_3, e_5
$u_2 = u^*$:
B_u $v_1, v_2, v_3, v_8, v_9, v_{10}$
λ_u e_2, e_8
B_u $v_3, v_4, v_5, v_6, v_9, v_{10}$
λ_u e_3, e_5
(a)
$u_0 = u$:
B_u $v_3, v_6, v_7, v_9, v_{10}$
λ_u e_2, e_6
u_1:
B_u $v_3, v_7, v_8, v_9, v_{10}$
λ_u e_3, e_7
B_u $v_3, v_4, v_5, v_6, v_9, v_{10}$
λ_u e_3, e_5
$u_2 = u^*$:
B_u $v_1, v_2, v_3, v_8, v_9, v_{10}$
λ_u e_2, e_8
(b)

Fig. 7. (a) non bag-maximal vs. (b) bag-maximal GHD of hypergraph H_0 in Figure 5.

Theorem 5.4. *For every hypergraph class $\mathscr{C}$ that enjoys the LogBMIP, and for every constant $k \geq 1$, the* Check*(GHD, k) problem is tractable, i.e., given a hypergraph H, it is feasible in polynomial time to check $ghw(H) \leq k$ and, if so, to compute a GHD of width k of H.*

Our plan is to make use of Theorem 4.5 by computing appropriate sets $\mathbf{S}$ of candidate bags such that $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$ if and only if there exists a GHD with width at most k. Example 5.5 illustrates the main challenge that needs to be tackled to compute such sets of candidate bags. A bag B_u in a GHD can be any subset of $B(\lambda_u)$ and choosing smaller subsets can decrease the width. At the same time, enumerating all subsets of $B(\lambda_u)$ is not an option if we are interested in classes of hypergraphs with unbounded rank. The main reason why the Check problem is tractable for HDs is that the additional special condition severely restricts the possible choices of B_u for given $B(\lambda_u)$.

Example 5.5 (Example 5.3 continued). In Figure 7, we have two GHDs of width 2 of the hypergraph H_0 from Figure 5. In the root u_0 of both GHDs, we have $v_2 \in B(\lambda_{u_0})$, since $v_2 \in e_2$ but $v_2 \notin B_{u_0}$. Hence, both GHDs violate the special condition in node u_0. However, if v_2 were added to B_{u_0}, then it can be seen that covering the edges e_1, e_2, e_7, e_8 below u_0 is no longer possible in a width 2 GHD. That is why the HD in Figure 6 has width 3. ◇

We start by introducing a useful property of GHDs, which we will call *bag-maximality*. Let $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ be a GHD of some hypergraph $H = (V(H), E(H))$. For each node u in T, we have $B_u \subseteq B(\lambda_u)$ by definition of GHDs and, in general, $B(\lambda_u) \setminus B_u$ may be non-empty. We observe that it is sometimes possible to take some vertices from $B(\lambda_u) \setminus B_u$ and add them to B_u without violating the connectedness condition. Of course, such an addition of vertices to B_u does not violate any of the other conditions of GHDs. Moreover, it does not increase the width.

Definition 5.6. Let $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ be a GHD of some hypergraph $H = (V(H), E(H))$. We call $\mathcal{H}$ *bag-maximal* if for every node u in T, adding a vertex $v \in B(\lambda_u) \setminus B_u$ to B_u would violate the connectedness condition.

It is easy to verify that if H has a GHD of width $\leq k$, then it also has a bag-maximal GHD of width $\leq k$. However, since we want to build on the algorithm from Section 4, we need to show that this also holds for bag-maximal ComNF GHDs. The problem here is that adding vertices to a bag B_u, to make it maximal, can change the set of $[B_u]$-components. Fortunately, we can reuse existing arguments on the existence of hypertree decompositions to show that if H has a GHD of width $\leq k$, then it indeed also has a bag-maximal ComNF GHD of width $\leq k$.

We now carry over several properties of HDs from Reference [31]. An inspection of the corresponding proofs in Reference [31] reveals that these properties hold also in the generalized case.

We thus state the following results below without explicitly "translating" the proofs of Reference [31] to the generalized setting. Note that Reference [31] deals with HDs and, therefore, in all decompositions considered there, the special condition holds. However, in Lemmas 5.7 and 5.8 below, the special condition is not needed.

We briefly recall the crucial notation for the following lemmas: For a set $V' \subseteq V(H)$, we define $nodes(V') = \{u \in T \mid B_u \cap V' \neq \emptyset\}$. If we want to make explicit the decomposition $\mathcal{G}$, then we also write $nodes(V', \mathcal{G})$ synonymously with $nodes(V')$. By further overloading the *nodes* operator, we also write $nodes(T_u)$ or $nodes(T_u, \mathcal{G})$ to denote the nodes in a subtree T_u of T, i.e., $nodes(T_u) = nodes(T_u, \mathcal{G}) = \{v \mid v \in T_u\}$.

Lemma 5.7 (Lemma 5.2 from [31]). *Consider an arbitrary GHD $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ of a hypergraph H. Let r be a node in T, let s be a child of r and let C be a $[B_r]$-component of H such that $C \cap V(T_s) \neq \emptyset$. Then, $nodes(C, \mathcal{H}) \subseteq nodes(T_s)$.*

Lemma 5.8 (Lemma 5.3 from [31]). *Consider an arbitrary GHD $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ of a hypergraph H. Let r be a node in T and let $U \subseteq V(H) \setminus B_r$ such that U is $[B_r]$-connected. Then $nodes(U, \mathcal{H})$ induces a (connected) subtree of T.*

Lemma 5.9. *For every GHD $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ of width k of a hypergraph H, there exists a bag-maximal ComNF GHD $\mathcal{H}'$ of H of width $\leq k$. Moreover, for every node u' in $\mathcal{H}'$ with cover $\lambda'_{u'}$, there exists a node u in $\mathcal{H}$ such that $\lambda_u = \lambda'_{u'}$.*

Proof. Start with a GHD $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ of width k of H. As long as there exists a node $u \in T$ and a vertex $v \in B(\lambda_u) \setminus B_u$, such that v can be added to B_u without destroying the GHD properties, select such a node u and vertex v arbitrarily and add v to B_u. By exhaustive application of this transformation, a bag-maximal GHD of width k of H is obtained.

We proceed by restating a procedure from Reference [31] that fixes violations of the ComNF condition: For the bag-maximal GHD obtained above, assume that there exist two nodes r and s such that s is a child of r, and the ComNF condition is violated for the pair, i.e., there does not exist a single $[B_r]$-component C_s such that $V(T_s) = C_s \cup (B_r \cap B_s)$. Let $C_1, \ldots, C_h$ be all the $[B_r]$-components containing some vertex occurring in $V(T_s)$. Hence, $V(T_s) \subseteq (\bigcup_{j=1}^{h} C_j \cup B_r)$. For each $[B_r]$-component C_j ($1 \leq j \leq h$), consider the set $nodes(C_j, \mathcal{H})$. By Lemma 5.8, $nodes(C_j, \mathcal{H})$ induces a subtree of T, and by Lemma 5.7, $nodes(C_j, \mathcal{H}) \subseteq nodes(T_s)$. Hence, $nodes(C_j, \mathcal{H})$ induces in fact a subtree of T_s.

For each node $n \in nodes(C_j, \mathcal{H})$ define a new node $u_{n,j}$ and let $\lambda_{u_{n,j}} = \lambda_n$ and $B_{u_{n,j}} = B_n \cap (C_j \cup B_r)$. Note that $B_{u_{n,j}} \neq \emptyset$, because by definition of $nodes(C_j, \mathcal{H})$, B_n contains some vertex belonging to C_j. Let $N_j = \{u_{n,j} \mid n \in nodes(C_j, \mathcal{H})\}$ and, for any C_j ($1 \leq j \leq h$), let T_j denote the (directed) graph (N_j, E_j) such that $u_{p,j}$ is a child of $u_{q,j}$ if and only if p is a child of q in T. T_j is clearly isomorphic to the subtree of T_s induced by $nodes(C_j, \mathcal{H})$, hence T_j is a tree as well.

Now transform the GHD $\mathcal{H}$ as follows: Delete the subtree T_s from T and attach to r every tree T_j for $1 \leq j \leq h$. In other words, we replace the subtree T_s by a set of trees $\{T_1, \ldots, T_h\}$. By construction, T_j contains a node $u_{n,j}$ for each node n belonging to $nodes(C_j, \mathcal{H})$ ($1 \leq j \leq h$). Then, if we let $children(r)$ denote the set of children of r in the new tree T obtained after the transformation above, then it holds that for any $s' \in children(r)$, there exists a $[B_r]$-component C of H such that $nodes(T_{s'}) = nodes(C, \mathcal{H})$, and $V(T_{s'}) \subseteq (C \cup B_r)$. We want to show $V(T_{s'}) = C \cup (B_{s'} \cap B_r)$. For the "$\supseteq$"-direction, we observe that $C \subseteq T_{s'}$ clearly holds, since we have $C \subseteq T_s$ and the bags $B_{u_{n,j}}$ in $T_{s'}$ were obtained from B_n in T_s as $B_{u_{n,j}} = B_n \cap (C_j \cup B_r)$ and we are considering the component $C = C_j$ here. Moreover, $B_{s'} \cap B_r \subseteq B_{s'} \subseteq V(T_{s'})$ clearly holds. Hence, we have $C \cup (B_{s'} \cap B_r) \subseteq V(T_{s'})$. For the "$\subseteq$"-direction, we conclude from $V(T_{s'}) \subseteq C \cup B_r$ that also $V(T_{s'}) \subseteq C \cup (V(T_{s'}) \cap B_r)$ holds. Hence, it suffices to show that $(V(T_{s'}) \cap B_r) \subseteq B_{s'}$ holds. By connectedness, we have $V(T_s) \cap B_r \subseteq B_s \cap B_r$ and, therefore, also $V(T_{s'}) \cap B_r \subseteq B_s \cap B_r$. Moreover, by construction, we have $B_{s'} = B_s \cap (C \cup B_r)$ and, therefore $B_s \cap B_r \subseteq B_{s'}$. We thus also arrive at $V(T_{s'}) \subseteq C \cup (B_{s'} \cap B_r)$.

It remains to show that $T_{s'}$ is bag-maximal. Assume to the contrary that there exists a node $u_{n,j} \in T_{s'}$ and a vertex $v \in B(\lambda_{u_{n,j}}) \setminus B_{u_{n,j}}$ such that v can be added to $B_{u_{n,j}}$ without destroying the GHD properties. Recall that $B_{u_{n,j}} = B_n \cap (C_j \cup B_r)$ and $\lambda_{u_{n,j}} = \lambda_n$. Since T was bag-maximal initially and from the construction (which only makes bags smaller), the only candidates for such a v are those vertices $B_n \setminus (C_j \cup B_r)$ that got removed from the bag. However, all neighboring nodes are either r or in $T_{s'}$, which means their bags are subsets of $C_j \cup B_r$. Hence, no $v \in B_n \setminus (C_j \cup B_r)$ is contained in a neighbor of $u_{n,j}$ and adding it would break connectedness. Our newly constructed $T_{s'}$ is therefore also bag-maximal.

Iterating this procedure for all ComNF violations will eventually produce a new GHD that is still bag-maximal and in ComNF. The second part of the statement follows from the observation that the construction never creates and new λ_u sets, but only reuses covers from the input decomposition. □

Example 5.10 (Example 5.5 continued). Clearly, the GHD in Figure 7(a) violates bag-maximality in node u', since the vertices v_4 and v_5 can be added to $B_{u'}$ without violating any GHD properties. If we add v_4 and v_5 to $B_{u'}$, then bag $B_{u'}$ at node u' and the bag at its child node are the same, which allows us to delete one of the nodes. This results in the GHD given in Figure 7(b), which is bag-maximal. In particular, the vertex v_2 cannot be added to B_{u_0}: Indeed, adding v_2 to B_{u_0} would violate the connectedness condition, since v_2 is not in B_{u_1} but in B_{u_2}. ◇

For the following arguments, the reader is advised to be careful in distinguishing between the bag B_u of a node u and the set of vertices $B(\lambda_u)$ that are covered by the integral edge cover λ_u. Before we prove a crucial lemma, we introduce some useful notation:

Definition 5.11. Let $\mathcal{H} = \langle T, (B_u)_{u\in T}, (\lambda_u)_{u\in T}\rangle$ be a GHD of a hypergraph H. Moreover, let u be a node in $\mathcal{H}$ and let $e \in \lambda_u$ such that $e \setminus B_u \neq \emptyset$ holds. Let u^* denote the node closest to u, such that u^* covers e, i.e., $e \subseteq B_{u*}$. Then, we call the path $\pi = (u_0, u_1, \dots, u_\ell)$ with $u_0 = u$ and $u_\ell = u^*$ the *critical path* of (u, e) denoted as *critp*(u, e).

Lemma 5.12. *Let $\mathcal{H} = \langle T, (B_u)_{u\in T}, (\lambda_u)_{u\in T}\rangle$ be a bag-maximal GHD of a hypergraph $H = (V(H), E(H))$, let $u \in T$, $e \in \lambda_u$, and $e \setminus B_u \neq \emptyset$. Let $\pi = (u_0, u_1, \dots, u_\ell)$ with $u_0 = u$ be the critical path of (u, e). Then the following equality holds:*

$$e \cap B_u = \quad e \cap \bigcap_{j=1}^{\ell} B(\lambda_{u_j}).$$

Proof. "$\subseteq$": Given that $e \subseteq B_{u_\ell}$ and by the connectedness condition, $e \cap B_u$ must be a subset of B_{u_j} for every $j \in \{1, \dots, \ell\}$. Therefore, $e \cap B_u \subseteq e \cap \bigcap_{j=1}^{\ell} B(\lambda_{u_j})$ holds.

"$\supseteq$": Assume to the contrary that there exists some vertex $v \in e$ with $v \notin B_u$ but $v \in \bigcap_{j=1}^{\ell} B(\lambda_{u_j})$. By $e \subseteq B_{u_\ell}$, we have $v \in B_{u_\ell}$. By the connectedness condition, along the path $u_0, \dots, u_\ell$ with $u_0 = u$, there exists $\alpha \in \{0, \dots, \ell - 1\}$, s.t. $v \notin B_{u_\alpha}$ and $v \in B_{u_{\alpha+1}}$. However, by the assumption, $v \in \bigcap_{j=1}^{\ell} B(\lambda_{u_j})$ holds. In particular, $v \in B(\lambda_{u_\alpha})$. Hence, we could safely add v to B_{u_α} without violating the connectedness condition nor any other GHD condition. This contradicts the bag-maximality of $\mathcal{H}$. □

Example 5.13 (Example 5.5 continued). Consider root node u of the GHD in Figure 7(b). We have $e_2 \in \lambda_u$ and $e_2 \setminus B_u = \{v_2\} \neq \emptyset$. As e_2 is covered by u_2, the critical path of (u, e_2) is $\pi = (u, u_1, u_2)$. It is easy to verify that $e_2 \cap B_u = e_2 \cap (e_3 \cup e_7) \cap (e_8 \cup e_2) = \{v_3, v_9\}$ indeed holds. ◇

ALGORITHM 1: `Union-of-Intersections-Tree`

```
input: An edge $e \in E(H)$, sets $Q_1, \dots, Q_\ell$ of edges of $E(H)$
output: $\bigcup\bigcap$-tree $T$ of $e, Q_1, \dots Q_\ell$
/* Initialization: compute $(N, E)$ for $T_0$                                   */
$N \leftarrow \{r\}$;
$E \leftarrow \emptyset$;
$label(r) \leftarrow \{e\}$;
$T \leftarrow (N, E)$;
/* Compute $T_j$ from $T_{j-1}$ in a loop over $j$                               */
for $j \leftarrow 1$ to $\ell$ do
    foreach leaf node $p$ of $T$ do
        if $label(p) \cap Q_j = \emptyset$ then
            Let $Q_j = \{e_{j1}, \dots, e_{jh_j}\}$;
            Create new nodes $\{p_1, \dots, p_{h_j}\}$;
            for $\alpha \leftarrow 1$ to $h_j$ do $label(p_\alpha) \leftarrow label(p_\alpha) \cup \{e_{j\alpha}\}$;
            $N \leftarrow N \cup \{p_1, \dots, p_{h_j}\}$;
            $E \leftarrow E \cup \{(p, p_1), \dots, (p, p_{h_j})\}$;
    $T \leftarrow (N, E)$;
```

Lemma 5.12 characterizes the overlap of an edge with a bag as an intersection of unions. However, to utilize the proposed intersection constraints, we would prefer unions of intersections instead. A straightforward transformation from an intersection of unions to a union of intersections may introduce certain redundant terms that we would like to avoid for technical reasons. We therefore employ a particular transformation, via the $\bigcup\bigcap$-trees defined below, that avoids such redundant terms in the union.

Definition 5.14. Let H be a hypergraph, e an edge of H and let $Q_1, \dots, Q_\ell$ be sets of edges. The $\bigcup\bigcap$-tree T of $e, Q_1 \dots, Q_\ell$ is the output of Algorithm 1 with inputs $e, Q_1, \dots Q_\ell$. We refer to the set of all leaves of T as $leaves(T)$.

Lemma 5.15. *Let H be a hypergraph, e an edge of H and let $Q_1, \dots, Q_\ell$ be sets of edges. Let T be the $\bigcup\bigcap$-tree of $e, Q_1, \dots, Q_\ell$, then*

$$e \cap \bigcap_{j=1}^{\ell} \bigcup Q_j = \bigcup_{p \in leaves(T)} \bigcap label(p).$$

Proof. Proof is by induction over ℓ. For $\ell = 0$, we have $leaves(T) = \{r\}$ and $label(r) = e$ and the statement trivially holds. For $0 \leq j \leq \ell$, let T_j denote the $\bigcup\bigcap$-tree of $e, Q_1, \dots, Q_j$. Suppose the statement is true for $\ell - 1$, then we observe the following equality:

$$e \cap \bigcap_{j=1}^{\ell} \bigcup Q_j = \left(e \cap \bigcap_{j=1}^{\ell-1} \bigcup Q_j \right) \cap \bigcup Q_\ell = \bigcup_{p \in leaves(T_{\ell-1})} \left(\bigcap label(p) \cap \bigcup Q_\ell \right),$$

where the right equality follows from the induction hypothesis and distribution of $\bigcup Q_\ell$ over the union over the leaves of $T_{\ell-1}$. Now, consider a leaf $p \in leaves(T_{\ell-1})$. The construction of T_ℓ either adds new leaves new_p as children of p, or p remains a leaf in T_ℓ. We claim that in the first case

$\bigcap label(p) \cap \bigcup Q_\ell = \bigcup_{p' \in new_p} \bigcap label(p')$ and in the second case, $\bigcap label(p) \cap \bigcup Q_\ell = \bigcap label(p)$. If the claim holds, then we have the following equality and the statement follows immediately:

$$\bigcup_{p \in leaves(T_{\ell-1})} \left(\bigcap label(p) \cap \bigcup Q_\ell\right) = \bigcup_{p \in leaves(T_\ell)} \bigcap label(p).$$

What is left is to verify the claim. The case where new children are added to p is straightforward by distributivity as the label of each new child corresponds to a term of the union $(\bigcap label(p) \cap e_{\ell 1}) \cup \cdots \cup (\bigcap label(p) \cap e_{\ell h_\ell}) = \bigcap label(p) \cap \bigcup Q_\ell$. If p remains a leaf, then we have $label(p) \cap Q_\ell \neq \emptyset$. Thus, $\bigcap label(p) \subseteq \bigcup Q_\ell$ and therefore $\bigcap label(p) \cap \bigcup Q_\ell = \bigcap label(p)$. □

Throughout the rest of this article, we will be interested in how bags can be represented as combinations of edges. In particular, we will see that, under the various restrictions introduced at the beginning of this section, we are able to bound the representation of bags as *unions of intersections* of edges. After introducing some notation for such unions of intersections, we can show how the LogBMIP allows for a bounded representation of bags for GHDs. The main result then follows by using this representation to compute an appropriate set of candidate bags, to which Algorithm 2 from Section 4 can then be applied.

Definition 5.16. Let H be a hypergraph. A *(q,p)-set* $X \subseteq V(H)$ is a set of the form $X = X_1 \cup \cdots \cup X_{q'}$ with $q' \leq q$ and where every X_i is the intersection of at most p edges. We will use q-set as shorthand for $(q,1)$-set.

We will repeatedly make use of the fact that, by the idempotence of union and intersection, we can w.l.o.g. assume a (q,p)-set to be the union of exactly q terms, each consisting of the intersection of exactly p edges.

Lemma 5.17. *Let H be a hypergraph with c-miwidth(H) $\leq i$ and let $B_1, \dots, B_\ell$ be $(q,1)$-sets. For each $e \in E(H)$, there exists a (q^{c-1}, c)-set I and a subedge $e' \subseteq e$ with $|e'| \leq i\, q^c$, such that*

$$e \cap \bigcap_{j=1}^{\ell} B_j = I \cup e'.$$

Furthermore, e' is the union of at most q^c subsets of intersections of exactly c edges.

Proof. For $j \in [\ell]$, fix a set of edges $Q_j = \{e_{j1}, \dots, e_{jq}\}$ such that $B_j = e_{j1} \cup \cdots \cup e_{jq}$. Let T be the $\bigcup\bigcap$-tree of $e, Q_1, \dots, Q_\ell$. For a node p of T, we refer to the number of edges in the path from the root to p as the depth of p, or $depth(p)$. Note that, by construction, $|label(p)| = depth(p) + 1$ for each node p in T. We consider the following partition of $leaves(T)$: Let $SMALL$ contain all the leaves of T at depth at most $c - 1$ and, conversely, let $FULL$ be the set of leaves at depth at least c.

By Lemma 5.15, $e \cap \bigcap_{j=1}^{\ell} B_j = (\bigcup_{p \in SMALL} \bigcap label(p)) \cup (\bigcup_{p \in FULL} \bigcap label(p))$. Hence, to prove the lemma, it suffices to show that $\bigcup_{p \in SMALL} \bigcap label(p)$ is a (q^{c-1}, c)-set and that $|\bigcup_{p \in FULL} \bigcap label(p)| \leq iq^c$ holds.

Claim A. *$\bigcup_{p \in SMALL} \bigcap label(p)$ is a (q^{c-1}, c)-set.*

Proof of Claim A. Since each of the sets Q_j has at most q members, every node in T has at most q children. Hence, there are at most q^{c-1} leaves at depth $\leq c - 1$ and therefore $|SMALL| \leq q^{c-1}$. Furthermore, we have $|label(p)| = depth(p) + 1 \leq c$, i.e., each intersection has at most c terms. ⋄

Claim B. *$|\bigcup_{p \in FULL} \bigcap label(p)| \leq iq^c$.*

Proof of Claim B. First, observe that for each $p \in FULL$, there exists a node p' in T at depth c such that $label(p) \supseteq label(p')$ and therefore also $\bigcap label(p) \subseteq \bigcap label(p')$. Note that there are at most q^c nodes p' at depth c. Furthermore, because $|label(p')| = c$ and we assume *c-miwidth(H) $\leq i$*, it holds that $|\bigcap label(p')| \leq i$. In total, we thus have that $\bigcup_{p \in FULL} \bigcap label(p)$ is a union of sets X_p such that each X_p is the subset of one out of at most q^c vertex sets, and each of these vertex sets has cardinality at most i. □

For a given edge cover λ_u and arbitrary edge $e \in E(H)$ with $\lambda_u(e) = 1$, Lemma 5.12 gives us a representation of $e \cap B_u$ of the form $e \cap \bigcap_{j=1}^{\ell} B(\lambda_{u_j})$. Clearly, the sets $B(\lambda_{u_j})$ are $(k, 1)$-sets, i.e., unions of (up to) k edges. We can therefore apply Lemma 5.17 by taking $B_j = B(\lambda_{u_j})$ and $q = k$ to get a representation of the form $I \cup e'$ for each of the possible subedges $e \cap B_u$ that may ever be used in a bag-maximal ComNF GHD. This idea is formalized in the following lemma, where we identify a polynomially big family of vertex sets $\mathbf{S} \subseteq 2^{V(H)}$, such that the bags of any bag-maximal ComNF GHD of H must be a member of this family:

Lemma 5.18. *Let H be a hypergraph with c-miwidth(H) $\leq b$, let $n = ||H||$ be the size of H, and fix an integer $k > 0$. There exists a set $\mathbf{S} \subseteq 2^{V(H)}$, which can be computed in $2^{b \cdot f_1(c,k)} n^{f_2(c,k)}$ time, such that* **ComCTD(S)** $\neq \emptyset$ *if and only if $ghw(H) \leq k$.*

Furthermore, for any bag-maximal ComNF GHD $\langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ of H of width $\leq k$, we have $\langle T, (B_u)_{u \in T} \rangle \in$ **ComCTD(S)**.

Proof. Let $m = |E(H)|$. We define the following sets:

$\mathcal{I} = \{I \mid I \text{ is a } (k^{c-1}, c)\text{-set}\}$,

$\mathcal{C} = \{e' \mid \text{there exist distinct } e_1, \dots, e_c \in E(H), \text{ such that } e' \subseteq e_1 \cap \dots \cap e_c\}$,

$Sub = \{I \cup \bigcup_{j=1}^{k^c} C_j \mid I \in \mathcal{I},\ C_1, \dots, C_{k^c} \in \mathcal{C}, \text{ and } I \cup \bigcup_{j=1}^{k^c} C_j \subseteq e \text{ for some } e \in E(H)\}$.

By construction, Sub contains only $E(H)$ and subedges of H. There are no more than $m^{ck^{c-1}}$ possible (k^{c-1}, c) sets. Also, by the condition *c-miwidth(H)* $\leq b$, we have $|\mathcal{C}| \leq 2^b m^c$ and, therefore, Sub has at most $m^{ck^{c-1}} m^{ck^c} 2^{bk^c}$ elements. Hence, Sub can be computed in $2^{b \cdot f(c,k)} n^{f'(c,k)}$) time for some functions f and f' by straightforward enumeration. We can now construct our desired set $\mathbf{S}$ as the set of all unions of up to k elements of Sub. It follows that the construction of $\mathbf{S}$ is possible in $2^{b \cdot f_1(c,k)} n^{f_2(c,k)}$ time for some functions f_1 and f_2.

It remains to show that this $\mathbf{S}$ indeed has the property that (1) **ComCTD(S)** $\neq \emptyset$ if and only if $ghw(H) \leq k$ and (2) for any bag-maximal ComNF GHD $\langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ of H of width $\leq k$, we have $\langle T, (B_u)_{u \in T} \rangle \in$ **ComCTD(S)**.

First, assume **ComCTD(S)** $\neq \emptyset$. Then there exists a TD of H where each bag is in $\mathbf{S}$ and therefore a union of k subedges of H. Hence, every bag of the TD can also be covered by k edges of H and thus can clearly be turned into a GHD of width at most k.

Now, assume $ghw(H) \leq k$. Let $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ be a bag-maximal ComNF GHD of width at most k and let u be a node of T. By Lemma 5.9, such a GHD always exists if $ghw(H) \leq k$. W.l.o.g., we assume that $B(\lambda_u) = e_1 \cup \dots \cup e_k$ and, therefore, also

$$B_u = B_u \cap B(\lambda_u) = (B_u \cap e_1) \cup \dots \cup (B_u \cap e_k).$$

We will show that $B_u \cap e_j \in Sub$ for each $j \in [k]$ and therefore also $B_u \in \mathbf{S}$. The case where $e_j \cap B_u = e_j$ is trivial as $e_j \in E(H) \subseteq Sub$. So, let e_j be any edge from this representation of $B(\lambda_u)$ where $e_j \cap B_u \neq e_j$. By Lemma 5.12, the following equality holds for the critical path $(u, u_1, \dots, u_\ell)$ of (u, e_j):

$$e_j \cap B_u = \quad e_j \cap \bigcap_{j=1}^{\ell} B(\lambda_{u_j}).$$

By assumption, every such $B(\lambda_{u_j})$ is a k-set and hence, by Lemma 5.17, we know that $e_j \cap \bigcap_{j=1}^{\ell} B(\lambda_{u_j})$ is precisely the union of an element of $\mathcal{I}$ and at most k^c sets from $\mathcal{C}$, i.e., $e_j \cap B_u \in$

Sub. Since the choice of u was arbitrary, every bag of $\mathcal{H}$ is contained in S and we have $\mathcal{H} \in$ **ComCTD**(S). □

Proof of Theorem 5.4: We assume that $\mathscr{C}$ enjoys the LogBMIP, i.e., for every $H \in \mathscr{C}$, we have *c-miwidth(H)* $\leq a \log n$. To solve the Check(GHD, k) problem, we can then simply compute, in polynomial time, the set S from Lemma 5.18 and decide whether **ComCTD**(S) $\neq \emptyset$. By Theorem 4.5 this is also tractable and, therefore, so is the whole procedure. □

We have defined in Section 1 the degree d of a hypergraph H. We now consider hypergraphs of bounded degree.

Definition 5.19. We say that a hypergraph H has the ***d*-bounded degree property (*d*-BDP)** if $degree(H) \leq d$ holds.

Let $\mathscr{C}$ be a class of hypergraphs. We say that $\mathscr{C}$ has the ***bounded degree property (BDP)*** if there exists a constant d such that every hypergraph H in $\mathscr{C}$ has the d-BDP.

The class of hypergraphs of bounded degree is an interesting special case of the class of hypergraphs enjoying the BMIP. Indeed, suppose that each vertex in a hypergraph H occurs in at most d edges for some constant d. Then the intersection of $d + 1$ hyperedges is always empty. The following corollary is thus immediate:

Corollary 5.20. *For every class $\mathscr{C}$ of hypergraphs of bounded degree, for each constant k, the problem Check(GHD, k) is tractable.*

In case of the BMIP, the upper bound on **S** in the proof of Lemma 5.18, is $2^{f(i,k,c)} m^{g(k,c)}$ for some function g. Recall from Theorem 4.5 that **ComCTD**(**S**)$\neq \emptyset$ can be decided in time complexity that is polynomial in $|\mathbf{S}| \cdot |V(H)|$. We thus get the following parameterized complexity result:

Theorem 5.21. *For constants k and c, the Check(GHD, k) problem is fixed-parameter tractable w.r.t. the parameter i for hypergraphs enjoying the ic-BMIP, i.e., in this case, Check(GHD, k) can be solved in time $O(h(i) \cdot poly(n))$, where $h(i)$ is a function depending on the intersection width i only and $poly(n)$ is a function that depends polynomially on the size $n = ||H||$ of the given hypergraph H.*

For practical purposes, this is of particular interest in case of the BIP, i.e., $c = 2$. Here, the construction of *Sub* can be significantly simplified, since, for any critical path $critp(u, e) = (u, u_1, \dots, u_\ell)$, it is enough to consider only one step, i.e., $e \cap B_u \subseteq e \cap B(\lambda_{\hat{u}}) = e \cap (e_{\hat{u}1} \cup \cdots \cup e_{\hat{u}k})$, for some node $\hat{u}$, such that all edges $e_{\hat{u}j}$ are distinct from e. This is the case for the following reasons: First, by connectedness, $e \cap B_u \subseteq e_{u_j1} \cup \cdots \cup e_{u_jk}$ for every node u_j on this critical path. And second, there is at least one node $\hat{u}$ on this critical path such that $e \notin \lambda_{\hat{u}}$. Indeed, if e were present in λ_{u_j} for every j, then we could add e to every bag B_{u_j} on the critical path without violating the connectedness condition. This would contradict the assumption of bag-maximality.

By the BIP, we have $|e \cap e_{\hat{u}j}| \leq i$ for every j and, therefore, we can simply compute all the subsets of $e \cap B(\lambda_{\hat{u}})$ in polynomial time to construct *Sub* as follows:

$$Sub = E(H) \cup \bigcup_{e \in E(H)} \left(\bigcup_{e_1, \dots, e_j \in (E(H) \setminus \{e\}),\, j \leq k} 2^{(e \cap (e_1 \cup \cdots \cup e_j))} \right),$$

i.e., *Sub* contains $E(H)$ and all subsets of intersections of edges $e \in E(H)$ with unions of $\leq k$ edges of H different from e. Here, the BIP is only used implicitly in the sense that we can be sure that the above expression is computable in polynomial time, because $|e \cap (e_1 \cup \cdots \cup e_j)| \leq ji \leq ki$ holds by the BIP. The explicit use of the BIP allows for a yet simpler (more coarse-grained) way to define *Sub* as follows:

$$Sub = E(H) \cup \{e' \mid e' \subseteq e \text{ for some } e \in E(H) \text{ and } |e'| \leq ki\}.$$

6 TRACTABLE CASES OF FHD COMPUTATION

6.1 Initial Considerations and Lemmas

In Section 5, we have shown that under certain conditions (with the BIP and BDP as most specific and the LogBMIP as most general conditions) the problem of computing a GHD of width $\leq k$ can be reduced to finding a ComNF candidate tree decomposition for an appropriate set of candidate bags. The key to this problem reduction was to enumerate a set of subedges, which allowed us to enumerate all bags of possible bag-maximal GHDs of width $\leq k$. When trying to carry over these ideas from GHDs to FHDs, we encounter *two major challenges*: Can we adapt our approach for GHDs to work with FHDs? And is it possible to find bounded representations of all the sets of vertices that can be *fractionally* covered with weight $\leq k$?

For the second challenge, recall from the GHD-case that the possible bags $B(\lambda_u)$ could be easily computed from the given set of subedges, since each λ_u can choose at most k subedges. In contrast, for a fractional cover γ_u, we do not have such a bound on the number of edges with non-zero weight. It is easy to exhibit a family $(H_n)_{n\in\mathbb{N}}$ of hypergraphs where it is advantageous to have unbounded $supp(\gamma_n)$ even if $(H_n)_{n\in\mathbb{N}}$ enjoys the BIP, as the following example illustrates:

Example 6.1. Consider the family $(H_n)_{n\in\mathbb{N}}$ of hypergraphs with $H_n = (V_n, E_n)$ defined as follows:

$V_n = \{v_0, v_1, \ldots, v_n\}$,
$E_n = \{\{v_0, v_i\} \mid 1 \leq i \leq n\} \cup \{\{v_1, \ldots, v_n\}\}$.

Clearly $iwidth(H_n) = 1$, but an optimal fractional edge cover of H_n is obtained by the following mapping γ with $supp(\gamma) = E_n$:

$\gamma(\{v_0, v_i\}) = 1/n$ *for each* $i \in \{1, \ldots, n\}$ *and*
$\gamma(\{v_1, \ldots, v_n\}) = 1 - (1/n)$

such that $weight(\gamma) = 2 - (1/n)$, which is optimal in this case.

Further below, we show that, for cases where the second challenge can be resolved (i.e., we can establish an upper bound on $supp(\gamma_u)$ for all nodes u in an FHD), the check-problem of FHDs can be essentially reduced to the GHD case. The respective Lemma 6.5 is thus the crucial tool for the remainder of this article. Before we move on to proving the crucial lemma, we need some further tools. Similarly as in the GHD-case, we will have to deal with unions of intersections of edges also in the FHD-case. The following definition and the accompanying two lemmas will be convenient for this purpose:

Definition 6.2. For a hypergraph H, we write $H^\cap$ for the closure of H under intersection of edges.

It will be important to observe that adding subedges does not change the fractional hypertree width of a hypergraph. Every FHD of H is still an FHD of $H^\cap$ and every FHD of $H^\cap$ can be easily transformed into an FHD of H. It follows that we always have $fhw(H) = fhw(H^\cap)$.

Lemma 6.3. *Let H be a hypergraph and γ a fractional edge cover of a set S of vertices. Then $B(\gamma)$ is a $2^{|supp(\gamma)|}$-set w.r.t. $H^\cap$.*

Proof. Let us call a subset $E \subseteq supp(\gamma)$ full if $\sum_{e\in E} \gamma(e) \geq 1$ and let **FS** contain all the full subsets. For every $E \in$ **FS**, we have $\bigcap E \subseteq B(\gamma)$ and it is straightforward to verify the following equality:

$$B(\gamma) = \bigcup_{E\in\mathbf{FS}} \bigcap E.$$

It is then enough to observe that $\bigcap E \in E(H^\cap)$ and that there are at most $2^{|supp(\gamma)|}$ full subsets. □

Lemma 6.4. *Let H be a hypergraph with c-miwidth(H) $\leq i$. Then $H^{\cap}$ can be computed in polynomial time for fixed c and i. Moreover, 2^c*-miwidth$(H^{\cap}) \leq i$ *holds.*

Proof. First, note that if $|E(H^{\cap})| \leq 2^c$, then also $|E(H)| \leq 2^c$ and we can trivially compute $H^{\cap}$ in polynomial time by simply computing all possible intersections of edges in $E(H)$. Moreover, in this case, the condition 2^c*-miwidth*$(H^{\cap}) \leq i$ is void, since there are no distinct 2^c edges in $H^{\cap}$.

Now let $\alpha = 2^c$ and consider an intersection I of α distinct edges of $H^{\cap}$ of the form $I = e'_1 \cap \cdots \cap e'_\alpha$, where $e'_j \in E(H^{\cap})$ for $j \in [\alpha]$. Each $e'_j \in E(H^{\cap})$ is an intersection of edges from $E(H)$, i.e., there exists $\mathcal{E}_j \subseteq E(H)$ such that $e'_j = \bigcap_{e \in \mathcal{E}_j} e$. Clearly, $I = \bigcap \left(\bigcup_{j=1}^{\alpha} \mathcal{E}_j \right)$.

We claim that $|\bigcup_{j=1}^{\alpha} \mathcal{E}_j| \geq c$, i.e., I is the intersection of at least c edges from $E(H)$. Indeed, with less than c distinct edges, there could only be less than $2^c - 1$ non-empty sets $\mathcal{E}_j$. It thus follows that I is a subset of an intersection of at least c edges of $E(H)$. Thus, by *c-miwidth(H)* $\leq i$, we conclude that $|I| \leq i$ holds. Hence, also in the case $|E(H^{\cap})| > 2^c$, the condition 2^c*-miwidth*$(H^{\cap}) \leq i$ holds.

It remains to show that $E(H^{\cap})$ can be computed from $E(H)$ in polynomial time: let $m = |E(H)|$. Then there are less than m^c intersections of less than c distinct edges from $E(H)$ and these intersections can clearly be computed in polynomial time. To compute also the set of intersections of at least c edges from $E(H)$, we proceed as follows: We first compute the set $\mathcal{I}_0$ of intersections of c edges. Then, for every $j \in [m - c]$, we compute the set $\mathcal{I}_j$ of intersections of α edges from $E(H)$ with $\alpha \in \{c, \ldots, c + j\}$. By *c-miwidth(H)* $\leq i$, we know that $|\mathcal{I}_j| \leq 2^i m^c$ holds for every $j \in [m - c]$. Hence, all these intersections can clearly be computed in polynomial time (for fixed c and i). □

Now that these properties of $H^{\cap}$ are verified, we can state the main technical lemma for this section. The main algorithmic results of this section are all based on the following statement:

Lemma 6.5. *Let H be a hypergraph with c-miwidth(H) $\leq i$ for constants c and i, and fix a rational k and integer $q \geq 1$. There exists a polynomial time algorithm that decides, whether there exists an FHD $\mathcal{F}$ of width k, such that for each node u of the decomposition with cover γ_u, it holds that $|supp(\gamma_u)| \leq q$.*

Proof. Let $\rho^*_{q,H}(X)$ denote the minimal weight of all fractional edge covers γ (in H) such that $X \subseteq B(\gamma)$ and $|supp(\gamma)| \leq q$. Using this function, we will define a set of candidate bags $\mathbf{S}$ such that $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$ if and only if there exists an FHD $\mathcal{F}$ as in the statement of the lemma. We demonstrate that $\mathbf{S}$ can be computed in polynomial time (for fixed c, i, k, and q) and the statement then follows from Theorem 4.5.

To define this set $\mathbf{S}$, we first use the result for the GHD-case: Let $\mathbf{S}_{ghd}$ be the set from Lemma 5.18 such that $\mathbf{ComCTD}(\mathbf{S}_{ghd}) \neq \emptyset$ if and only if $ghd(H^{\cap}) \leq 2^q$. By Lemma 6.4, $H^{\cap}$ can be computed in polynomial time, and thus by Lemma 5.18, we have that such an $\mathbf{S}_{ghd}$ exists and can also be computed in polynomial time. We can then obtain the set $\mathbf{S} = \{S \in \mathbf{S}_{ghd} \mid \rho^*_{q,H}(S) \leq k\}$. The function $\rho^*_{q,H}(S)$ can be computed, for each set $S \in \mathbf{S}_{ghd}$, by solving a linear program for every combination of q edges in $E(H)$, i.e., by solving $\binom{|E(H)|}{q}$ linear programs. Hence, also $\mathbf{S}$ can be computed in polynomial time.

We now argue that $\mathbf{ComCTD}(\mathbf{S})$ behaves as claimed in the beginning of the proof. Suppose $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$, and recall that $H^{\cap}$ has the same vertices as H. Then clearly there is a TD where every bag has $\rho^*_{q,H}$ at most k and, therefore, there is a width k FHD where every cover has support bounded by q.

For the other direction, suppose $\mathcal{F} = \langle T, (B_u)_{u \in T}, (\gamma_u)_{u \in T} \rangle$ is an FHD of H with width k, such that for each node u of $\mathcal{F}$ with cover γ_u, it holds that $|supp(\gamma_u)| \leq q$. By Lemma 6.3, there exists an FHD $\mathcal{F}' = \langle T, (B_u)_{u \in T}, (\gamma'_u)_{u \in T} \rangle$ of $H^{\cap}$ with the same width, such that $B(\gamma'_u) = B(\gamma_u)$ and $B(\gamma'_u)$ is a 2^q-set.

That is, each $B(\gamma'_u)$ is of the form $e_{u,1} \cup \cdots \cup e_{u,\ell_u}$ with $\ell_u \leq 2^q$. Now let $\lambda_u = \{e_{u,1}, \ldots, e_{u,\ell_u}\}$ for each $u \in T$. It is easy to see that $\mathcal{H} = \langle T, (B_u)_{u \in T}, (\lambda_u)_{u \in T} \rangle$ is a GHD of $H^\cap$ with $ghw(\mathcal{H}) \leq 2^q$. Note that $\mathcal{H}$ is not necessarily bag-maximal or in ComNF. However, following the procedure described in Lemma 5.9, there exists a bag-maximal ComNF $\mathcal{H}' = \langle T', (B'_u)_{u \in T'}, (\lambda'_u)_{u \in T'} \rangle$ of $H^\cap$ with $ghw(\mathcal{H}') \leq 2^q$ and therefore $\langle T', (B'_u)_{u \in T'} \rangle \in$ **ComCTD**$(\mathbf{S}_{ghd})$.

Now recall that the transformation into a bag-maximal ComNF $\mathcal{H}'$ from Lemma 5.9 never changes the covers in the GHD, i.e., if λ is the cover of some node in $\mathcal{H}'$, then that same λ already was a cover of some node in $\mathcal{H}$. That is, every bag of $\mathcal{H}'$ is a subset of set $\bigcup \lambda_u$ for some node u in $\mathcal{H}$. Recall that we have $\bigcup \lambda_u = B(\gamma'_u)$ where γ'_u is the fractional cover in $\mathcal{F}'$. Hence, every bag of $\mathcal{H}'$ is also a subset of some $B(\gamma_u)$ in $\mathcal{F}$. Therefore, $\langle T', (B'_u)_{u \in T'} \rangle$is in ComNF, each of its bags is in $\mathbf{S}_{ghd}$ and can be fractionally covered (in H) with weight k and at most q edges in the support, i.e., $\langle T', (B'_u)_{u \in T'} \rangle \in$ **ComCTD**(**S**). □

To apply Lemma 6.5, we will have to prove that an appropriate constant q indeed exists. The following two subsections will show that such a constant indeed exists for classes of either bounded intersection width or bounded degree. Furthermore, in Section 7, we will also use the lemma to derive an approximation result for classes that exhibit the BMIP.

6.2 Computing FHDs under Bounded Degree

First, we will investigate the situation for classes of hypergraphs whose degree is bounded by a constant d. We show that a fractional edge cover γ of a hypergraph H has support $supp(\gamma)$ bounded by a constant that depends only on $weight(\gamma)$ and d. From there, tractability follows from Lemma 6.5.

Theorem 6.6. *For every hypergraph class $\mathscr{C}$ that has bounded degree, and for every rational constant $k \geq 1$, the* Check*(FHD, k) problem is tractable, i.e., given a hypergraph $H \in \mathscr{C}$, it is feasible in polynomial time to check $fhw(H) \leq k$ and, if so, to compute an FHD of width k of H.*

For this result, we need to introduce, analogously to edge-weight functions and edge covers in Section 2, the notions of vertex-weight functions and vertex covers.

Definition 6.7. A vertex-weight function w for a hypergraph H assigns a weight $w(v) \geq 0$ to each vertex v of H. We say that w is a *fractional vertex cover* of H if for each edge $e \in E(H)$, $\Sigma_{v \in e} w(v) \geq 1$ holds. For a vertex-weight function w for hypergraph H, we denote by $weight(w)$ its total weight, i.e., $\Sigma_{v \in V(H)} w(v)$. The *fractional vertex cover number* $\tau^*(H)$ is defined as the minimum $weight(w)$ where w ranges over all fractional vertex covers of H. The *vertex support* $vsupp(w)$ of a hypergraph H under a vertex-weight function w is defined as $vsupp(w) = \{v \in V(H) \mid w(v) > 0\}$.

For our result on bounded support, we will exploit the well-known dualities $\rho^*(H) = \tau^*(H^d)$ and $\tau^*(H) = \rho^*(H^d)$, where H^d denotes the dual of H. To make optimal use of this, we make, for the moment, several assumptions. First, we will assume that (1) hypergraphs have no isolated vertices and (2) no empty edges. Furthermore, we assume that (3) hypergraphs never have two distinct vertices of the same "edge-type" (i.e., we exclude that two vertices occur in precisely the same edges) and (4) they never have two distinct edges of the same "vertex-type" (i.e., we exclude duplicate edges).

Assumptions (1)–(4) can be safely made. Recall that we are ultimately interested in the computation of an FHD of width $\leq k$ for given k. As mentioned above, without assumption (1), the computation of an edge-weight function and, hence, of an FHD of width $\leq k$ makes no sense. Assumption (2) does not restrict the search for a specific FHD, since we would never define an edge-weight function with non-zero weight on an empty edge. As far as assumption (3) is concerned, suppose that a hypergraph H has groups of multiple vertices of identical edge-type. Then it is sufficient to consider the reduced hypergraph H^- resulting from H by "fusing" each such group to a single vertex. Obviously $\rho^*(H) = \rho^*(H^-)$, and each edge-weight function for H^- can be extended in the obvious way to an edge-weight function of the same total weight to H. Finally, assumption (4) can also be made w.l.o.g., since we can again define a reduced hypergraph H^- resulting from H by retaining only one edge from each group of identical edges. Then every edge cover of H^- is an edge cover of H. Conversely, every edge cover of H can be turned into an edge cover of H^- by assigning to each edge e in H^- the sum of the weights of e and all edges identical to e in H.

Under our above assumptions (1)–(4), for every hypergraph H, the property $H^{dd} = H$ holds and there is an obvious one-to-one correspondence between the edges (vertices) of H and the vertices (edges) of H^d. Moreover, there is an obvious one-to-one correspondence between the fractional edge covers of H and the fractional vertex covers of H^d. In particular, if there is a fractional edge cover γ for H, then its corresponding "dual" γ^d assigns to each vertex v of H^d the same weight as to the edge in H that is represented by this vertex and vice versa.

Note that if we do not make assumptions (3) and (4), then there are hypergraphs H with $H^{dd} \neq H$. For instance, consider the hypergraph H_0 with $V(H_0) = \{a, b, c\}$ and $E(H_0) = \{\, e = \{a, b, c\}\,\}$, i.e., property (3) is violated. The hypergraph H_0^d has a unique vertex e and a unique hyperedge $\{e\}$. Hence, H_0^{dd} is (isomorphic to) the hypergraph with a unique vertex a and a unique hyperedge $\{a\}$, which is clearly different from the original hypergraph H_0.

To get an upper bound on the support $supp(\gamma)$ of a fractional edge cover of a hypergraph H, we make use of the following result for fractional vertex covers. This result is due to Zoltán Füredi [26], who extended earlier results by Chung et al. [16]. Below, we appropriately reformulate Füredi's result for our purposes:

Proposition 6.8 ([26], page 152, Proposition 5.11.(iii)). *For every hypergraph H of rank (i.e., maximal edge size) r, and every fractional vertex cover w for H satisfying $weight(w) = \tau^*(H)$, the property $|vsupp(w)| \leq r \cdot \tau^*(H)$ holds.*

By duality, exploiting the relationship $\rho^*(H) = \tau^*(H^d)$ and by recalling that the degree of H corresponds to the rank of H^d, we immediately get the following corollary:

Corollary 6.9. *For every hypergraph H of degree d, and every fractional edge cover γ for H satisfying $weight(\gamma) = \rho^*(H)$, the property $|supp(\gamma)| \leq d \cdot \rho^*(H)$ holds.*

Proof of Theorem 6.6. Let $\mathscr{C}$ be a class of hypergraphs with degree at most d and let $H \in \mathscr{C}$. Observe that we can extend Corollary 6.9 to any vertex subset $V' \subseteq V(H)$ because $\rho^*(V')$ in H is the same as $\rho^*(H[V'])$. The degree of the induced subhypergraph can not become greater than the degree of H.

Therefore, for every set of vertices $V' \subseteq V(H)$, if $\rho^*(V') \leq k$, then V' can be fractionally covered using at most kd edges. Thus, $fhw(H) \leq k$ if and only if there exists a width k FHD of H where the edge cover γ_u of every node u has $|supp(\gamma_u)| \leq kd$. By Lemma 6.5, this can be decided in polynomial time. □

6.3 Computing FHDs under Bounded Intersection

We will now follow the same strategy used in the previous section on bounded degree to prove tractability of checking *fhw* for hypergraph classes enjoying the BIP. However, bounded intersection width has different structural implications than bounded degree. Example 6.1 illustrated our main challenge in the fractional setting, namely, the potentially unbounded size of the support of an optimal fractional edge cover. In that example, the growth of the support is linked to the increase in the degree of v_0, whereas the intersection width remains 1, no matter how large n

becomes. A combinatorial result that bounds the size of the support in terms of the optimal weight of the cover and the intersection width is therefore impossible.

Instead, we will show that if the optimal weight of a fractional cover γ is below some fixed bound k, then we can find another fractional cover ν of weight $\leq k$ that can be expressed by a combination of edges and vertices, such that the number of both is bounded by functions of k and the intersection width i. Analogously to the case of bounded degree, we will then derive tractability of the CHECK(FHD, k) problem via Lemma 6.5, which relies on the bounded support of ν.

Our proof ultimately relies on two key observations:

(1) If all edges in a fractional cover γ have weight bounded by some constant, then also the size of $B(\gamma)$ is bounded by some constant. In Lemma 6.10, we will make the relationship between the bound on the weights of individual edges and the bound on the size of $B(\gamma)$ precise.
(2) If an edge has weight close to 1 in an optimal cover, then we can in fact set this weight to 1 at the expense of a negligible increase of the weight. In Lemma 6.13, we will show under which circumstances we may increase the weight of an individual edge and still stay below k with the total weight of γ.

We start by formalizing the first observation mentioned above.

LEMMA 6.10. *Let $k \geq 1$, $i \in \mathbb{N}$, and $\delta > 0$ be constants, let H be a hypergraph with iwidth$(H) \leq i$ and let γ be an edge-weight function of H with weight$(\gamma) \leq k$. Moreover, let $\gamma(e) \leq 1 - \delta$ for every edge e. Then $|B(\gamma)| \leq \frac{ik^2}{\delta}$ holds.*

PROOF. For an arbitrary edge $e \in supp(\gamma)$, let $m_e = |B(\gamma) \cap e|$. We first show that $m_e \leq \frac{ik}{\delta}$. Indeed, we are assuming that $\gamma(e) \leq 1 - \delta$ holds. That is, e puts weight $\leq 1 - \delta$ on each vertex in $B(\gamma) \cap e$. Hence, weight $\geq \delta$ has to be put on each vertex in $B(\gamma) \cap e$ by the other edges. In total, the other edges thus have to put weight $\geq m_e\delta$ on the m_e vertices in $B(\gamma) \cap e$.

By the upper bound i on the intersection width, whenever γ puts weight w on some edge e' different from e, then, in total, at most weight iw is put on the vertices in e. Since we are assuming $weight(\gamma) \leq k$, the total weight of all edges in $E(H)$ is $\leq k$. Hence, in total, at most weight ik can be put on the vertices in $B(\gamma) \cap e$ by the edges different from e. We therefore have $m_e\delta \leq ik$ or, equivalently, $m_e \leq \frac{ik}{\delta}$.

Now let m be the maximum size of $B(\gamma) \cap e$ attainable for any edge $e \in supp(\gamma)$. Moreover, for an arbitrary edge $e \in supp(\gamma)$, suppose that $\gamma(e) = w$ holds. Then, in total, e puts weight $\leq mw$ on the vertices in $B(\gamma)$. Hence, the total weight put by all edges of $supp(\gamma)$ on all vertices in $B(\gamma)$ is $\leq mk$. Moreover, every vertex in $B(\gamma)$ receives weight at least 1. Hence, $|B(\gamma)| \leq mk$. Together with the above upper bound $\frac{ik}{\delta}$ on the size of $B(\gamma) \cap e$ for every edge e, we thus get $|B(\gamma)| \leq \frac{ik^2}{\delta}$. □

Our next goal is to show that every fractional cover γ can be replaced by a fractional cover that is, in a sense, very close to an integral cover. To formalize this closeness to an integral cover, we introduce the notion of *c-bounded fractional part*.

Definition 6.11. For a hypergraph H, edge weight function $\gamma : E(H) \to [0, 1]$, and $E' \subseteq supp(\gamma)$, we write $\gamma|_{E'}$ to denote the *restriction of γ to E'*, i.e., $\gamma|_{E'}(e) = \gamma(e)$ if $e \in E'$ and $\gamma|_{E'}(e) = 0$ otherwise.

Now let $E' = \{e \in supp(\gamma_u) \mid \gamma_u(e) < 1\}$ and let $c \in \mathbb{N}$. We say that γ has *c-bounded fractional part*, if $|B(\gamma_u|_{E'})| \leq c$.

In Lemma 6.13, we will formalize also the second observation from above. To this end, we first introduce the following terminology, which will be useful in the proof of Lemma 6.13:

Definition 6.12. For $d \in \mathbb{N}$ and rational $k \geq 1$, we define:

$$\begin{aligned} \mathcal{H}(d) &= \{H = (V, E) \mid |V| \leq d\}, \\ M(k, d) &= \{\rho^*(H) - k \mid H \in \mathcal{H}(d) \text{ and } \rho^*(H) > k\}, \\ \mu(k, d) &= \min(M(k, d)), \text{ if } M(k, d) \neq \emptyset \text{ and undefined otherwise.} \end{aligned}$$

Lemma 6.13. *Let $k \geq 0$ be a rational number and $i \in \mathbb{N}$. Moreover, let H be a hypergraph with $iwidth(H) \leq i$ and let $S \subseteq V(H)$ with $S \neq \emptyset$. Then there exists a positive rational $\epsilon(k, i)$ and an integer $f(k, i)$ such that the following holds: If $\rho^*(S) \leq k + \epsilon(k, i)$, then $\rho^*(S) \leq k$ and there exists an edge weight function $\nu\colon E(H) \to [0, 1]$ with the following properties:*

- *$S \subseteq B(\nu)$,*
- *$weight(\nu) \leq k$, and*
- *ν has c-bounded fractional part with $c = f(k, i)$.*

Proof. The proof is by induction on k. For $0 \leq k < 1$, we set $\epsilon(k, i) = \frac{1-k}{2}$ and $f(k, i) = 0$. Then the statement vacuously holds, since we have $k + \epsilon(k, i) < 1$ and there cannot be $S \neq \emptyset$ with $\rho^*(S) < 1$. For $k \geq 1$, we define

$$\begin{aligned} \delta &= \frac{\epsilon(k-1, i).}{2}, \\ d &= \frac{ik^2}{\delta} = \frac{2ik^2}{\epsilon(k-1, i).}, \\ f(k, i) &= \max(d, f(k-1, i)), \end{aligned}$$

and we choose $\epsilon(k, i)$ with $\epsilon(k, i) < \frac{\epsilon(k-1,i).}{2}$ and, if $\mu(k, d)$ is defined, also $\epsilon(k, i) < \mu(k, d)$.

Now consider $S \neq \emptyset$ with $\rho^*(S) \leq k + \epsilon(k, i)$ and let γ be an optimal fractional cover of S, i.e., $weight(\gamma) = \rho^*(S)$ and $S \subseteq B(\gamma)$. We distinguish two cases:

Case 1. Suppose that there exists $e \in supp(\gamma)$ with $\gamma(e) > 1 - \delta$. Consider $W = S \setminus e$. Then, we have

$$\begin{aligned} \rho^*(W) &\leq \rho^*(S) - \gamma(e) \leq k + \epsilon(k, i) - (1 - \delta) = k + \epsilon(k, i) - 1 + \frac{\epsilon(k-1, i).}{2} \leq \\ &\leq k + \frac{\epsilon(k-1, i).}{2} - 1 + \frac{\epsilon(k-1, i).}{2} = (k-1) + \epsilon(k-1, i). \end{aligned}$$

Thus, by the induction hypothesis, there exists an edge weight function $\nu_W\colon E(H) \to [0, 1]$ with the following properties:

- $W \subseteq B(\nu_W)$,
- $weight(\nu_W) \leq k - 1$, and
- ν_W has c_w-bounded fractional part with $c_W = f(k-1, i)$.

Then, we can extend ν_W to the desired edge cover ν of S by setting $\nu(e) = 1$ and $\nu(e') = \nu_W(e')$ for every edge e' with $e' \neq e$.

Case 2. Suppose that $\gamma(e) \leq 1 - \delta$ for every $e \in supp(\gamma)$. Then, by Lemma 6.10, $|B(\gamma)| \leq \frac{ik^2}{\delta}$. We claim that γ is in fact the desired edge cover of S with c-bounded fractional part for $c = f(k, i)$. Indeed, by the above definitions of δ, d, and $f(k, i)$, we have $|B(\gamma)| \leq \frac{ik^2}{\delta}$ and $f(k, i) \geq d$ with $d = \frac{ik^2}{\delta} = \frac{2ik^2}{\epsilon(k-1,i).}$. Hence, γ clearly has c-bounded fractional part with $c = f(k, i)$. It only remains to show that $weight(\gamma) \leq k$. Suppose to the contrary that $weight(\gamma) > k$. Note that the subhypergraph of H induced by S is in $\mathcal{H}(d)$ with $d = \frac{ik^2}{\delta} = \frac{2ik^2}{\epsilon(k-1,i).}$ and γ is an optimal fractional edge cover of S. Hence, $M(k,d) = \{\rho^*(H) - k \mid H \in \mathcal{H}(d) \text{ and } \rho^*(H) > k\}$ is non-empty and, therefore, $\mu(k,d) = \min(M(k,d))$ is defined. Moreover, $\rho^*(S) > k$ implies $\rho^*(S) \geq k + \mu(k,d)$. However, we have chosen $\epsilon(k,i)$ with $\epsilon(k,i) < \mu(k,d)$. Hence, from $\rho^*(S) \leq k + \epsilon(k,i)$ it follows that $\rho^*(S) < k + \mu(k,d)$ holds, a contradiction. □

If an edge cover γ of weight $\leq k$ has c-bounded fractional part, then $B(\gamma)$ is of the form $B(\gamma) = e_1 \cup \cdots \cup e_\ell \cup U$ with $\ell \leq k$, $|U| \leq c$, and $\{e \in E \mid \gamma(e) = 1\} = \{e_1, \ldots, e_\ell\}$. W.l.o.g., we may assume $U \cap e_j = \emptyset$ for every j. Hence, there are at most 2^c "significantly different" edges in $supp(\gamma)$ as far as covering U is concerned, that is $|\{e \cap U \mid e \in supp(\gamma)\}| \leq 2^c$. Hence, γ can be replaced by an edge weight function γ' with $weight(\gamma') = weight(\gamma)$, $B(\gamma) \subseteq B(\gamma')$ and γ' has support bounded by $k + 2^c$. We thus immediately obtain the following corollary, which may be of independent interest:

Corollary 6.14. *Let H be a hypergraph with $iwidth(H) \leq i$ and let $\gamma \colon E(H) \to [0,1]$ with $weight(\gamma) \leq k$. Then there exists an assignment $\nu \colon E(H) \to [0,1]$ such that $weight(\nu) \leq k$, $B(\gamma) \subseteq B(\nu)$ and ν has bounded support (depending on i and k).*

Analogously to the case of bounded degree dealt with in Theorem 6.6, we now combine Corollary 6.14 with Lemma 6.5 to prove the main theorem of this subsection:

Theorem 6.15. *For every hypergraph class $\mathscr{C}$ that enjoys the BIP, and for every rational constant $k \geq 1$, the Check(FHD, k) problem is tractable, i.e., given a hypergraph $H \in \mathscr{C}$, it is feasible in polynomial time to check $fhw(H) \leq k$ and, if so, to compute an FHD of width k of H.*

Proof. Let $\mathscr{C}$ be a class of hypergraphs with intersection width at most i and let $H \in \mathscr{C}$. By Corollary 6.14, there exists a constant q, such that $fhw(H) \leq k$ holds if and only if there exists an FHD of width at most k where for every node u, the cover γ_u has $|supp(\gamma_u)| \leq q$. Thus, we can again decide $fhw(H) \leq k$ via the bounded support case, which is tractable according to Lemma 6.5. □

Deciding the Check Problem for Hypergraphs of Bounded Rank. We conclude this section by discussing an important special case of bounded intersection width: bounded rank. It is easy to see that for every hypergraph H, we have $iwidth(H) \leq rank(H)$. The tractability of the Check(FHD,k) problem for classes of bounded rank is therefore already a consequence of Theorem 6.15. However, the complexity of the algorithm described above may be prohibitive for practical use. We therefore choose to briefly present a significantly simpler method for hypergraph classes with bounded rank.

Definition 6.16. We say that a hypergraph H has the ***r-bounded rank property (r-BRP)*** if $rank(H) \leq r$ holds. For a class $\mathscr{C}$ of hypergraphs, we say that $\mathscr{C}$ has the ***bounded rank property (BRP)*** if there exists a constant r such that every hypergraph H in $\mathscr{C}$ has the r-BRP.

For every hypergraph with $rank(H) \leq r$ for some constant r, the following lemma is immediate:

Lemma 6.17. *Let H be a hypergraph whose rank is bounded by some constant r. Then for every (fractional) edge weight function γ for H satisfying $weight(\gamma) \leq k$, the property $|B(\gamma)| \leq r \cdot k$ holds.*

We can now use the above lemma to directly generate the candidate bags for the algorithm from Theorem 4.5. For any hypergraph H, we simply compute the set $\mathbf{S}$ of all sets with at most $r \cdot k$ vertices that can be fractionally covered with weight k. Clearly, if $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$, then there exists an FHD of width k. However, if $fhw(H) \leq k$, then there exists a ComNF FHD $\langle T, (B_u)_{u \in T}, (\gamma_u)_{u \in T} \rangle$ of width k. This is easy to see from the proof of Lemma 5.9. The construction of a ComNF GHD there translates directly to FHDs. Hence, $\langle T, (B_u)_{u \in T} \rangle$ is a ComNF TD where every bag is in $\mathbf{S}$, i.e., $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$. We thus get the following special case of Theorem 6.15:

Corollary 6.18. *For every hypergraph class $\mathscr{C}$ that has bounded rank, and for every rational constant $k \geq 1$, the* Check(FHD, k) *problem is tractable, i.e., given a hypergraph $H \in \mathscr{C}$, it is feasible in polynomial time to check $fhw(H) \leq k$ and, if so, to compute an FHD of width k of H.*

7 APPROXIMATION OF *fhw* UNDER CONSTRAINTS

We now turn our attention to approximations of the *fhw*. It is known from Reference [44] that a tractable cubic approximation of the *fhw* always exists, i.e.: for $k \geq 1$, there exists a polynomial-time algorithm that, given a hypergraph H with $fhw(H) \leq k$, finds an FHD of H of width $O(k^3)$. In this section, we search for conditions that guarantee a better approximation of the *fhw*.

The natural first candidate for restricting hypergraphs is the BMIP. For the Check(GHD, k) problem, this restriction guarantees tractability. Furthermore, the BIP and the BDP also lead to tractability of the Check(FHD, k) problem for fixed $k \geq 1$. Both of these properties are generalized by the BMIP, making it the natural next step. For the fractional case, we will show that a significantly better polynomial-time approximation of the *fhw* than in the general case is possible for hypergraphs enjoying the BMIP. It is noteworthy that we can reuse the strategy from the exact cases without modifications for this task. Finally, we show that classes with bounded **Vapnik-Chervonenkis (VC)** dimension—an even more general property than the BMIP—also allow for a polynomial-time approximation of the *fhw*, but with slightly weaker guarantees.

7.1 Approximation of *fhw* under BMIP

We first inspect the case of the bounded multi-intersection property. We will show that the BMIP allows for an arbitrarily close approximation of the *fhw* in polynomial time. Formally, the main result of this section is as follows:

Theorem 7.1. *Let $\mathscr{C}$ be a hypergraph class that enjoys the BMIP and let k, ϵ be arbitrary rational constants with $k \geq 1$ and $\epsilon > 0$. Then there exists a polynomial-time algorithm that, given a hypergraph $H \in \mathscr{C}$ with $fhw(H) \leq k$, finds an FHD of H of width $\leq k(1 + \epsilon)$.*

Even though we are now interested in approximations, we can still employ the same strategy that was used in the previous section. However, it is not clear how to find a bounded representation of the sets that can be covered with weight k. Yet, if we introduce an arbitrarily small constant $\epsilon > 0$, then it is possible to find supersets with weight $k(1 + \epsilon)$ that are indeed representable in the desired fashion, as we will show in Lemma 7.3 below. The proof of this lemma will make use of the following combinatorial result:

Lemma 7.2. *Fix an integer $c \geq 1$. Let $X = (x_1, \ldots, x_n)$ be a sequence of positive numbers $\leq \delta$ and fix w such that $\sum_{j=1}^{n} x_j \geq w \geq \delta c$. Then, we have $\sum x_{i_1} \cdot x_{i_2} \cdot \cdots \cdot x_{i_c} \geq (w - \delta c)^c$, where the sum is over all c-tuples $(i_1, \ldots, i_c)$ of* distinct *integers from $[n]$.*

Proof. We proceed by induction over c. For $c = 1$ the statement holds by the requirement that $\sum_{j=1}^{n} x_j \geq w$. Suppose the statement holds for $c - 1$. Let us consider only the tuples with a fixed value for i_c and let us consider only the sum over those tuples, which we can write as $x_{i_c} \cdot \sum x_{i_1} \cdot x_{i_2} \cdot \cdots \cdot x_{i_{c-1}}$. Notably, the sum now ranges only over all $(c - 1)$-tuples of distinct integers from $[n] \setminus \{i_c\}$. Therefore, we need to apply the induction hypothesis for the values in X, except for x_{i_c}. In particular, we apply the hypothesis with $w' := w - \delta$ instead of w as the lower bound in the premise. The lower bound w' for the sum over the values in X, except for x_{i_c}, holds, since $x_{i_c} \leq \delta$ by assumption. Applying the induction hypothesis in this way then gives us the following inequality:

$$x_{i_c} \cdot \sum x_{i_1} \cdot x_{i_2} \cdot \cdots \cdot x_{i_{c-1}} \geq x_{i_c}((w - \delta) - \delta(c - 1))^{c-1} = x_{i_c}(w - \delta c)^{c-1}.$$

Now, take the sum over all values in $[n]$ for i_c to again consider all the c-tuples, and we arrive at the statement:

$$\sum x_{i_1} \cdot x_{i_2} \cdot \dots \cdot x_{i_c} \geq (w - \delta c)^{c-1} \sum_{j=1}^{n} x_j \geq w(w - \delta c)^{c-1} \geq (w - \delta c)^c. \qquad \square$$

Lemma 7.3. *Let H be a hypergraph with c-miwidth(H) $\leq i$ for some $c \geq 1$ and $i \geq 0$ such that H is closed under intersection of edges and contains all edges of size 1. Moreover, fix a $k \geq 1$ and an ϵ with $0 < \epsilon \leq 1$. Then there exists an integer q, depending only on k, c, i, and ϵ, such that for any set of vertices S with $\rho^*(S) \leq k$, there exists a q-set S' with $S \subseteq S'$ and $\rho^*(S') \leq k(1 + \epsilon)$.*

Proof. Let γ be a fractional edge cover of some $S \subseteq V(H)$ with $weight(\gamma) \leq k$. Similar to the proof for Check(FHD, k) under BIP, we will partition the edges with non-zero weight into heavy and light edges. This time the threshold for heavy edges is chosen to be $\epsilon/(4c)$, i.e., the sets E_ℓ and E_h are defined as follows:

$E_\ell = \{e \in supp(\gamma) \mid \gamma(e) < \epsilon/(4c)\}$,
$E_h = \{e \in supp(\gamma) \mid \gamma(e) \geq \epsilon/(4c)\}$.

Let X be the set of all vertices for which the weight of $\gamma|_{E_h}$ is at least $1 - \epsilon/2$ and let $S' = S \cup X$. We will show that S' has the desired properties by showing that X and $S \setminus X$ are q_1 and q_2 sets for appropriately chosen q_1 and q_2, both of which depend only on k, c, i, and ϵ.

First, we can increase the weight of each edge in γ by the factor $(1 + \epsilon)$ to obtain a new γ' with $S' \subseteq B(\gamma')$ and $weight(\gamma') \leq k(1 + \epsilon)$. Now, in γ', X is fully covered by the edges in E_h, i.e., $X \subseteq B(\gamma'|_{E_h})$. This is due to the fact that $(1 + \epsilon)(1 - \epsilon/2) = 1 + \epsilon/2 - \epsilon^2/2 \geq 1$ holds by the assumption $0 < \epsilon \leq 1$, i.e., if $\gamma|_{E_h}$ put at least $1 - \epsilon/2$ weight on X, then $\gamma'|_{E_h}$ will put at least weight 1 on X. Since we are assuming $weight(\gamma) \leq k$ and $\gamma(e) \geq \epsilon/(4c)$ for every $e \in E_h$, there can be no more than $4ck/\epsilon$ edges in E_h. From Lemma 6.3, we then know that X is a $\left(2^{4ck/\epsilon}\right)$-set.

To complete the proof, we show that $S \setminus X$ contains at most $f(c, k, i, \epsilon)$ vertices and can therefore be represented as a union of that many edges of size 1. By the definition of X, the vertices in $S \setminus X$ need to receive at least $\epsilon/2$ weight from edges in E_ℓ. As every light edge contributes $\leq \epsilon/(4c)$ weight, there need to be at least $2c$ light edges incident to every vertex in $S \setminus X$.

We proceed with a counting argument. Imagine a bipartite graph $G = (S \setminus X, T, E(G))$ where T is the set of all c-tuples of distinct light edges. There is an edge from $v \in S \setminus X$ to $(e_1, \dots, e_c) \in T$ iff v is in $e_1 \cap \dots \cap e_c$. Furthermore, we assign weight $\prod_{j=1}^{c} \gamma(e_j)$ to every edge in $E(G)$ incident to a $(e_1, \dots, e_c) \in T$.

We now count the total weight in G from both sides. First observe that on the T side, we have degree at most i because c-*miwidth(H)* $\leq$ i. Therefore, the total weight is at most $i \cdot \sum_{(e_1, \dots, e_c) \in T} \prod_{j=1}^{c} \gamma(e_j)$. Observe that $\sum_{(e_1, \dots, e_c) \in T} \prod_{j=1}^{c} \gamma(e_j) \leq \left(\sum_{e_1 \in E_\ell} \gamma(e_1)\right) \cdot \dots \cdot \left(\sum_{e_c \in E_\ell} \gamma(e_c)\right)$ as, by distributivity, all the terms of the left-hand side sum are also present on the right-hand side of the inequality. Furthermore, we have $\sum_{e \in E_\ell} \gamma(e) \leq k$ and thus, by putting it all together, we see that the total weight in G is at most $k^c i$.

From the $S \setminus X$ side, consider an arbitrary vertex $v \in S \setminus X$ and let $e_1, \dots, e_n$ be the light edges in $E(H)$ containing v. As v is not in X, the heavy edges of γ contribute less than $1 - \epsilon/2$ weight to v. Hence, we have $\sum_{j=1}^{n} \gamma(e_j) \geq \epsilon/2$ and $\gamma(e_j) < \epsilon/(4c)$ for each $j \in [n]$. We can apply Lemma 7.2 for $X = \{\gamma(e_1), \dots, \gamma(e_n)\}$, $\delta = \epsilon/(4c)$, and $w = \epsilon/2$ to get the inequality $\sum \gamma(e_{j_1}) \cdot \dots \cdot \gamma(e_{j_c}) \geq (\epsilon/2 - c\epsilon/(4c))^c = (\epsilon/4)^c$, where the sum ranges over all c-tuples $(e_{j_1}, \dots, e_{j_c})$ of distinct edges from $\{e_1, \dots, e_n\}$, which denote the light edges in $E(H)$ containing v.

We conclude that v (now considered as a vertex in G) is incident to edges whose total weight is $\geq (\epsilon/4)^c$ in $E(G)$. Since we have seen above that the total weight of all edges in $E(G)$ is $\leq k^c i$, there can be no more than $i(4k/\epsilon)^c$ vertices in $S \setminus X$. □

Proof of Theorem 7.1. We proceed analogously to the proofs of the two tractability results for *fhw* in Section 6. Recall that, for every $H \in \mathscr{C}$, we are assuming *c-miwidth(H)* $\leq i$ for constants c and i. Let H be a hypergraph in $\mathscr{C}$ and let H' be the new hypergraph obtained by computing $H^\cap$ and adding every edge of size 1. As argued before, $\mathit{fhw}(H') = \mathit{fhw}(H)$ and, by Lemma 6.4, we can compute H' in polynomial time.

From the combination of Lemma 7.3 with Lemma 6.5, we have that there exists a polynomial time computable set **S** with $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$ if and only if $\mathit{fhw}(H') \leq k(1+\epsilon)$. We can then use Theorem 4.5 to decide $\mathbf{ComCTD}(\mathbf{S}) \neq \emptyset$ and, in consequence, $\mathit{fhw}(H') \leq k(1+\epsilon)$ and, therefore, also $\mathit{fhw}(H) \leq k(1+\epsilon)$. □

A polynomial time approximation scheme for finding optimal FHDs. Recall the definition of the K-Bounded-FHW-Optimization problem from Section 1, i.e.: Given a hypergraph H, we are interested in $\mathit{fhw}(H)$, but only if $\mathit{fhw}(H) \leq K$. We will now show that, with the algorithm from Theorem 7.1 at our disposal, we are able to give a polynomial time approximation scheme for the bounded optimization problem. More precisely, we aim at an approximation algorithm with the following properties:

Definition 7.4 (PTAS [9, 53]). Let Π be an (intractable) minimization problem with positive objective function f_Π. An algorithm `Alg` is called an *approximation scheme* for Π if on input (I, ϵ), where I is an instance of Π and $\epsilon > 0$ is an error parameter, it outputs a solution s such that:

$$f_\Pi(I, s) \leq (1+\epsilon) \cdot f_\Pi(I, s^*),$$

where s^* is an optimal solution of I, i.e., for all other solutions s' of I it holds that $f_\Pi(I, s^*) \leq f_\Pi(I, s')$.

`Alg` is called a ***polynomial time approximation scheme*** **(PTAS)** if for every fixed $\epsilon > 0$, its running time is bounded by a polynomial in the size of instance I.

We now show that, in case of the BMIP, the K-Bounded-FHW-Optimization problem indeed allows for a PTAS:

Theorem 7.5. *For every hypergraph class $\mathscr{C}$ that enjoys the BMIP, there exists a PTAS for the K-Bounded-FHW-Optimization problem.*

Proof. By Theorem 7.1, there exists a function `find-fhd` (H, k, ϵ, c, i) with the following properties:

- `find-fhd` takes as input a hypergraph H with *c-miwidth(H)* $\leq i$ and numbers $k \geq 1$, $\epsilon \geq 0$;
- `find-fhd` returns an FHD $\mathcal{F}$ of H of width $\leq k + \epsilon$ if $\mathit{fhw}(H) \leq k$ or `fail`. In the latter case it is guaranteed that $\mathit{fhw}(H) > k$. Note that finding an FHD of width $\leq k + \epsilon$ is equivalent to finding an FHD of width $\leq k(1+\epsilon')$ for $\epsilon' = \epsilon/k$.
- `find-fhd` runs in time polynomial in the size of H, where k, ϵ, c, and i are considered as fixed.

We can then obtain the required PTAS by calling `find-fhd` $(H, k', \epsilon/2, c, i)$ for every width $k' = a \cdot \frac{\epsilon}{2} + 1$ such that $1 \leq k' \leq K + \epsilon/2$ and a is a non-negative integer. The procedure returns the lowest k' (and the respective FHD) for which `find-fhd` accepts. Since this requires no more than $2K/\epsilon$ calls to `find-fhd`, the procedure is clearly polynomial.

It is also not difficult to verify the correctness of the procedure. If $\mathit{fhw}(H) \leq K$, then `find-fhd` is called for a width k', such that $\mathit{fhw}(H) \leq k' \leq \mathit{fhw}(H) + \epsilon/2$. By Theorem 7.1, this call returns an FHD $\mathcal{F}$ width at most $k' + \epsilon/2 \leq fhw(H) + \epsilon$. However, suppose our procedure returns width k' and the FHD $\mathcal{F}$ with width at most $k' + \epsilon/2$. By definition, k' is the lowest width for which `find-fhd` accepted, hence `find-fhd` rejected for the next lowest parameter $k' - \epsilon/2$. Therefore, we can deduce that $k' - \epsilon/2 \leq fhw(H)$ and thus also $k' \leq fhw(H) + \epsilon/2$. □

7.2 Approximation of *fhw* under Bounded VC Dimension

We now present a polynomial-time approximation of the *fhw* for classes of hypergraphs enjoying bounded **Vapnik-Chervonenkis (VC)** dimension. This bounded VC dimension is yet again more general property than BMIP but the approximation guarantees in this class will be slightly weaker. We will combine some classical results on the VC dimension with some novel observations. This will yield an approximation of the *fhw* up to a logarithmic factor. We first recall the definition of the VC-dimension of hypergraphs.

Definition 7.6 ([49, 52]). Let $H = (V(H), E(H))$ be a hypergraph and $X \subseteq V(H)$ a set of vertices. Denote by $E(H)|_X$ the set $E(H)|_X = \{X \cap e \mid e \in E(H)\}$. The vertex set X is called *shattered* if $E(H)|_X = 2^X$. The *VC dimension* $\mathrm{vc}(H)$ of H is the maximum cardinality of a shattered subset of $V(H)$.

We now provide a link between the VC-dimension and our approximation of the *fhw*.

Definition 7.7. Let $H = (V(H), E(H))$ be a hypergraph. A *transversal* (also known as *hitting set*) of H is a subset $S \subseteq V(H)$ that has a non-empty intersection with every edge of H. The *transversality* $\tau(H)$ of H is the minimum cardinality of all transversals of H.

Clearly, $\tau(H)$ corresponds to the minimum of the following integer linear program: Find a mapping $w : V \rightarrow \{0, 1\}$ that minimizes $\Sigma_{v \in V(H)} w(v)$ under the condition that $\Sigma_{v \in e} w(v) \geq 1$ holds for each hyperedge $e \in E$.

The *fractional transversality* τ^* of H is defined as the minimum of the above linear program when dropping the integrality condition, thus allowing mappings $w : V \rightarrow \mathbb{R}_{\geq 0}$. Finally, the *transversal integrality gap tigap*(H) of H is the ratio $\tau(H)/\tau^*(H)$.

Recall that computing the mapping λ_u for some node u in a GHD can be seen as searching for a minimal edge cover ρ of the vertex set B_u, whereas computing γ_u in an FHD corresponds to the search for a minimal fractional edge cover ρ^* [35]. Again, these problems can be cast as linear programs where the first problem has the integrality condition and the second one has not. Further, we can define the *cover integrality gap cigap*(H) of H as the ratio $\rho(H)/\rho^*(H)$. With this, we state the following approximation result for *fhw*:

Theorem 7.8. *Let $\mathscr{C}$ be a class of hypergraphs with VC-dimension bounded by some constant d and let $k \geq 1$. Then there exists a polynomial-time algorithm that, given a hypergraph $H \in \mathscr{C}$ with $fhw(H) \leq k$, finds an FHD of H of width $O(k \cdot \log k)$.*

Proof. The proof proceeds in several steps.

Reduced hypergraphs. Recall the notion of reduced hypergraphs from Section 6.2. There, it was established that we can assume, w.l.o.g., that every hypergraph is reduced. For the rest of this proof, we therefore consider only reduced hypergraphs. This ensures that $(H^d)^d = H$ holds. It is well-known and easy to verify that the following relationships between H and H^d hold for any reduced hypergraph H, (see, e.g., Reference [21]):

(1) The edge coverings of H and the transversals of H^d coincide.
(2) The fractional edge coverings of H and the fractional transversals of H^d coincide.
(3) $\rho(H) = \tau(H^d)$, $\rho^*(H) = \tau^*(H^d)$, and $cigap(H) = tigap(H^d)$.

VC-dimension. By a classical result (Reference [20], Theorem (5.4); see also Reference [12] for related results), for every hypergraph $H = (V(H), E(H))$ with at least two edges, we have:

$$tigap(H) = \tau(H)/\tau^*(H) \leq 2\text{vc}(H)\log(11\tau^*(H)).$$

For hypergraphs H with a single edge only, $\text{vc}(H) = 0$, and thus the above inequation does not hold. However, for such hypergraphs $\tau(H) = \tau^*(H) = 1$. By putting this together, we get:

$$tigap(H) = \tau(H)/\tau^*(H) \leq \max(1, 2\text{vc}(H)\log(11\tau^*(H))).$$

Moreover, in Reference [7], it is shown that $\text{vc}(H^d) < 2^{\text{vc}(H)+1}$ always holds. In total, we thus get

$$\begin{aligned} cigap(H) = tigap(H^d) &\leq \max(1, 2\text{vc}(H^d)\log(11\tau^*(H^d))) \\ &\leq \max(1, 2^{\text{vc}(H)+2}\log(11\rho^*(H))) \\ &\leq \max(1, 2^{d+2}\log(11\rho^*(H))), \text{ which is } O(\log\rho^*(H)). \end{aligned}$$

Approximation of fhw by ghw. Suppose that H has an FHD $\langle T, (B_u)_{u\in V(T)}, (\lambda)_{u\in V(T)}\rangle$ of width k. Then there exists a GHD of H of width $O(k \cdot \log k)$. Indeed, we can find such a GHD by leaving the tree structure T and the bags B_u for every node u in T unchanged and replacing each fractional edge cover γ_u of B_u by an optimal integral edge cover λ_u of B_u. By the above inequality, we thus increase the weight at each node u only by a factor $O(\log k)$. Moreover, we know from Reference [4] that $hw(H) \leq 3 \cdot ghw(H) + 1$ holds. In other words, we can compute an HD of H (which is a special case of an FHD) in polynomial time, whose width is $O(k \cdot \log k)$. □

To conclude, the following Lemma 7.9 establishes a relationship between BMIP and VC-dimension:

Lemma 7.9. *If a class $\mathscr{C}$ of hypergraphs has the BMIP, then it has bounded VC-dimension. However, there exist classes $\mathscr{C}$ of hypergraphs with bounded VC-dimension that do not have the BMIP.*

Proof. [BMIP ⇒ bounded VC-dimension.] Let $c \geq 1, i \geq 0$ and let H be a hypergraph with *c-miwidth(H)* $\leq i$. We claim that then $\text{vc}(H) \leq c + i$ holds.

Assume to the contrary that there exists a set $X \subseteq V$, such that X is shattered and $|X| > c + i$. We pick c arbitrary, pairwise distinct vertices $v_1, \dots, v_c$ from X and define $X_j = X \setminus \{v_j\}$ for each j. Then $X = (X_1 \cap \dots \cap X_c) \cup \{v_1, \dots, v_c\}$ holds and also $|X| \leq |X^*| + c$ with $X^* \subseteq X_1 \cap \dots \cap X_c$.

Since X is shattered, for each $1 \leq j \leq c$, there exists a distinct edge $e_j \in E(H)$ with $X_j = X \cap e_j$. Hence, $X_j = X \setminus \{v_j\} \subseteq e_j$ and also $X^* \subseteq e_1 \cap e_2 \cap \dots \cap e_c$ holds, i.e., X^* is in the intersection of c edges of H. By *c-miwidth(H)* $\leq i$, we thus get $|X^*| \leq i$. In total, we have $|X| \leq |X^*| + c \leq i + c$, which contradicts our assumption that $|X| > c + i$ holds.

[bounded VC-dimension ⇏ BMIP.] It suffices to exhibit a family $(H_n)_{n\in\mathbb{N}}$ of hypergraphs such that $\text{vc}(H_n)$ is bounded, whereas *c-miwidth(H_n)* is unbounded for any constant c. We define $H_n = (V_n, E_n)$ as follows:

$V_n = \{v_1, \dots, v_n\}$,
$E_n = \{V_n \setminus \{v_i\} \mid 1 \leq i \leq n\}$.

Clearly, $\text{vc}(H_n) < 2$. Indeed, take an arbitrary set $X \subseteq V$ with $|X| \geq 2$. Then $\emptyset \subseteq X$ but $\emptyset \neq X \cap e$ for any $e \in E_n$. However, let $c \geq 1$ be an arbitrary constant and let $X = e_{i_1} \cap \dots \cap e_{i_\ell}$ for some $\ell \leq c$ and edges $e_{i_j} \in E_n$. Obviously, $|X| \geq n - c$ holds. Hence, also *c-miwidth(H_n)* $\geq n - c$, i.e., it is not bounded by any constant $i \geq 0$. □

In the first part of Lemma 7.9, we have shown that $\text{vc}(H) \leq c+i$ holds. For an approximation of an FHD by a GHD, we need to approximate the fractional edge cover γ_u of each bag B_u by an integral edge cover λ_u, i.e., we consider fractional vs. integral edge covers of the induced hypergraphs

$H_u = (B_u, E_u)$ with $E_u = \{e \cap B_u \mid e \in E(H)\}$. Obviously, the bound $\text{vc}(H) \leq c + i$ carries over to $\text{vc}(H_u) \leq c + i$.

8 CONCLUSION AND FUTURE WORK

In this work, we have settled the complexity of deciding $\mathit{fhw}(H) \leq k$ for fixed constant $k \geq 2$ and $\mathit{ghw}(H) \leq k$ for $k = 2$ by proving the NP-completeness of both problems. This gives negative answers to two open problems. On the positive side, we have identified rather mild restrictions such as the BDP (i.e., the bounded degree property), BIP (i.e., the bounded intersection property), LogBIP, BMIP (i.e., the bounded multi-intersection property), and LogBMIP, which give rise to a Ptime algorithm for the Check(GHD, k) problem. Moreover, we have shown that the BDP and the BIP ensure tractability also of the Check(FHD, k) problem. For the BMIP, we have shown that an arbitrarily close approximation of the *fhw* in polynomial time exists. In case of bounded VC dimension, we have proposed a polynomial-time algorithm for approximating the *fhw* up to a logarithmic factor. As the empirical analyses reported in Reference [23] show, these restrictions are very well-suited for instances of CSPs and, even more so, of CQs. We believe that they deserve further attention.

Our work does not finish here. We plan to explore several further issues regarding the computation and approximation of the fractional hypertree width. We find the following questions particularly appealing: (i) Does the special condition defined by Grohe and Marx [35] lead to tractable recognizability also for FHDs, i.e., in case we define "*sc-fhw*(H)" as the smallest width an FHD of H satisfying the special condition, can $\mathit{sc}\text{-}\mathit{fhw}(H) \leq k$ be recognized efficiently? (ii) Our tractability result in Section 5 for the Check(FHD, k) problem is weaker than for Check(GHD, k). In particular, for the BMIP, we have only obtained efficient approximations of the *fhw*. Note that the techniques applied in the tractability proof of Check(FHD, k) in case of the BIP do not carry over in an obvious way to the BMIP. An important property in our tractability proof is that edges with low weight can only contribute a *constant* number of vertices to $B(\gamma)$. This property crucially depends on the BIP, which implies that the additional weight put by other edges on these vertices is bounded by ki. Between submission and publication of this manuscript, we were indeed able to generalize the tractability result for Check(FHD, k) to hold also under the BMIP. We refer to the respective publication for further details [30].

APPENDIX

A A POLYNOMIAL TIME ALGORITHM FOR THE COMNF CTD PROBLEM

In this section, we provide a proof of Theorem 4.5. To do so, we present a bottom-up construction of ComNF CTDs, if they exist, using bottom-up dynamic programming. Note that our presentation does not optimize for runtime; our goal is only to establish that the problem can be decided in polynomial time. As before, for a hypergraph H and $U \subseteq V(H)$, we write $H[U]$ for the subhypergraph induced by the vertices U.

Definition A.1. A pair (B, C) of *disjoint* subsets of $V(H)$ is a *block* if C is a $[B]$-component of H or $C = \emptyset$. Such a block is *headed* by B. Let (B, C) and (X, Y) be two blocks. We say that $(X, Y) \leq (B, C)$ if $X \cup Y \subseteq B \cup C$ and $Y \subseteq C$.

In a bottom-up search for a decomposition, we can consider a block to correspond to a subproblem. Namely, a block (B, C) corresponds to the subproblem of whether there exists a ComNF decomposition of $H[B \cup C]$, where the root has bag B. If such a decomposition exists, then we call a block *satisfied*. Note that if $C = \emptyset$, the block is trivially satisfied. Our overall goal will be to find a set B, such that all blocks headed by B are satisfied. In the context of finding candidate TDs, we will eventually consider only those blocks that are headed by a candidate bag.

Lemma A.2. *Let H be a hypergraph and $B \subseteq V(H)$. If all blocks headed by B are satisfied, then H has a ComNF tree decomposition where B is the bag of the root.*

Proof. Let $(B, C_1), \dots, (B, C_\ell)$ be all the blocks headed by B. Since each of the blocks is satisfied, there exist ComNF TDs $\langle T_1, (B_{1,u})_{u \in T_1} \rangle, \dots, \langle T_\ell, (B_{\ell,u})_{u \in T_\ell} \rangle$, respectively. Furthermore, the root of each such TD has B as its bag.

The construction of a TD of H is then straightforward by simply *merging* the individual trees at their common root node with bag B. It is staightforward to verify that this is indeed a valid TD. Any edge $e \in E(H)$ is a subset of $B \cup C_i$ for some i, since each block corresponds to a separate $[B]$-component. Thus, e is already covered in the corresponding subtree. Furthermore, for distinct $i, j \in [\ell]$, we have that C_i and C_j are disjoint. Hence, any vertices that are shared by subtrees for two blocks are in B and the connectedness therefore follows from the connectedness of each subtree. □

In standard dynamic programming fashion, our algorithm will decide whether a block (B, C) is satisfied by checking whether certain subproblems are satisfied. Intuitively, these subproblems are the blocks that correspond to the subtrees under the root of the decomposition for $H[B \cup C]$. However, because we will proceed bottom-up, some additional technical considerations need to be made to guarantee the tree decomposition properties. We capture this in the following notion of a *basis* of a block:

Definition A.3. For a block (B, C) and vertex set $B' \subseteq V(H)$ with $B' \neq B$ and let $(B', C'_1), \dots, (B', C'_\ell)$ be all the blocks headed by B' such that $(B', C'_i) \leq (B, C)$ for all $i \in [\ell]$. We say that B' *is a basis of* (B, C) if the following conditions hold:

(1) $C \subseteq B' \cup \bigcup_{i=1}^{\ell} C'_i$,
(2) For each $e \in E(H)$ such that $e \cap C \neq \emptyset$, $e \subseteq B' \cup \bigcup_{i=1}^{\ell} C'_i$,
(3) For each $i \in [\ell]$, the block (B', C'_i) is satisfied.

The following two lemmas now show that this notion of basis indeed strongly corresponds to the existence of a tree decomposition. First, we show that if a block has a basis, then it is satisfied, demonstrating that the check can indeed be reduced to satisfiedness of subproblems. The second lemma shows that in any ComNF TD, child nodes form a basis of the block corresponding to their subtree. After these two lemmas, we are ready to state the full algorithm.

Lemma A.4. *Let (B, C) be a block and B' a basis of (B, C). Then (B, C) is satisfied.*

Proof. Let (B', C'_i) for some $i > 1$ be as in Definition A.3. For the sake of a simpler presentation, we will write C' for the set $B' \cup \bigcup_{i=1}^{\ell} C'_i$. First, observe that by the assumption in the statement and by Lemma A.2 there exists a ComNF TD $\mathcal{T} = \langle T, (B_u)_{u \in T} \rangle$ of $H[C']$ where B' is the bag of the root node.

Consider the TD $\mathcal{T}'$ obtained by adding a new root node u_0 with bag B to $\mathcal{T}$ and attaching the old root as the only child of u_0. We claim that $\mathcal{T}'$ is a ComNF TD of $H[B \cup C]$, and thus that (B, C) is satisfied.

To verify this claim, we first consider connectedness in $\mathcal{T}'$. We know that the connectedness condition holds in $\mathcal{T}$. By construction, the only way the condition can fail in $\mathcal{T}'$ is if there is a $v \in \bigcup_{i=1}^{\ell} C'_i$ that is also in B. Suppose v is in a C'_i, then it is also in C because $(B', C'_i) \leq (B, C)$. But C and B are disjoint, hence, $v \notin B$ and $\mathcal{T}'$ satisfies the connectedness condition.

We move on to showing that every edge of $H[B \cup C]$ is covered in $\mathcal{T}'$. By the second condition of a basis, any $e \in E(H)$ is either fully covered by C', or disjoint with C. If we consider an edge $e' = e \cap (B \cup C)$ in $H[B \cup C]$, then we can distinguish two cases. If $e' \cap C \neq \emptyset$, then $e' \subseteq C'$ and thus e' (and actually even e itself) is covered already in $\mathcal{T}$. Otherwise, if $e' \cap C = \emptyset$, then $e' = e \cap B$. Thus, e' is covered in B.

Finally, since all the blocks (B', C'_i) are $\leq (B, C)$, we have $C' \subseteq B \cup C$. In particular, because we assume $C \subseteq C'$, we also have $B \cup C' = B \cup C$. Thus, $V(\mathcal{T}') = B \cup C$ and the claim holds. □

Lemma A.5. *Let H be a hypergraph and $\langle T, (B_u)_{u \in T} \rangle$ be a ComNF TD of H. Let $r \in T$ be a non-leaf node. For each child s of r, let C_s be the $[B_r]$-component associated with s. The following two statements are true:*

- *$(B_s, D) \leq (B_r, C_s)$ if and only if D is either a component associated with a child of s, or if $D = \emptyset$,*
- *B_s is a basis of the block (B_r, C_s).*

Proof. We first observe that $(B_s, D) \leq (B_r, C_s)$ if and only if D is either empty or a component associated with a child of s. Indeed, since we assume ComNF, we have that $V(T_s) = C_s \cup (B_r \cap B_s)$ and C_s is the only such $[B_r]$-component. Thus, $V(T_s) \subseteq B_r \cup C_s$.

Moreover, $B_s \cup D \subseteq V(T_s)$ holds for every block (B_s, D) where D is empty or a component associated with a child of s, which completes the proof of the "if" direction. For the "only if" direction, recall that $(B_s, D) \leq (B_r, C_s)$ requires $D \subseteq C_s$. Since there is only one node associated with C_s, we conclude $D \subseteq V(T_s)$. Since D is a $[B_s]$-component, it must have its own associated child of s.

Now that we know exactly which blocks headed by B_s are relevant, we can show that they satisfy the conditions of a basis. Let $child(s)$ be the set of all children of s. For every $u \in child(s)$, let D_u be the $[B_s]$-component associated with u. The vertices that occur in the subtree T_s are precisely $V(T_s) = B_s \cup \bigcup_{u \in child(s)} D_u$. Since we assume ComNF, we also have $C_s \subseteq V(T_s)$ and therefore Condition 1 of a basis is satisfied. Furthermore, it is straightforward to verify that for every $u \in child(s)$, the subtree rooted at node s with only T_u attached below it, is a ComNF TD of $H[B_s \cup D_u]$. Hence, the block (B_s, D_u) is satsified for every such u and Condition 3 holds.

For Condition 2 it is enough to observe that if $e \cap C_s \neq \emptyset$, then it must be covered in the subtree T_s, i.e., $e \subseteq V(T_s)$. Otherwise, suppose e were only covered in some node u not in the subtree T_s. There is a vertex $v \in e \cap C_s$ that occurs in $V(T_s)$ but not in B_r (recall B_r and C_s are disjoint). Any path from a node of T_s to u must pass through B_r, which would break connectedness for v. □

Algorithm 2 describes the resulting procedure. We simply build, bottom-up, a list of satisfied blocks. The check whether *"B' is a basis of (B, C)"* is to be understood as relative to the program state. That is, Condition 3 of a basis in Definition A.3 is checked w.r.t. whether the blocks are marked as satisfied in the algorithm.

Proof of Theorem 4.5. We only present the decision procedure. Soundness of the procedure follows immediately by Lemmas A.4 and A.2. Furthermore, it follows from the soundness argument that constructing an appropriate TD from an accepting state is trivial. Completeness of the procedure follows by Lemma A.5, which demonstrates that any $\mathcal{T} \in \mathbf{ComCTD}(\mathbf{S})$ can be found in this bottom-up fashion.

What is left to show is that Algorithm 2 indeed always terminated in polynomial time. Observe that a straightforward representation of $\mathbf{S}$ as a list of lists of vertices has size $O(|\mathbf{S}| \cdot |V(H)| \cdot \log |V(H)|)$. For asserting polynomial runtime it is therefore not necessary to distinguish between the size of the representation of $\mathbf{S}$ and $|\mathbf{S}|$. The set *blocks* has at most $|\mathbf{S}| \cdot |V(H)|$ elements and computing components is polynomial in the size of the representation of H. Furthermore, when we consider Condition 3 to refer to blocks that are marked as satisfied, deciding whether a set B' is a basis is also polynomial. Thus, the whole algorithm requires only polynomial time. □

ALGORITHM 2: ComNF Candidate Tree Decomposition

```
input: Hypergraph $H$ and a set $\mathsf{S} \subseteq 2^{V(H)}$.
output: “Accept,” if $\mathbf{ComCTD}(\mathsf{S}) \neq \emptyset$
        “Reject,” otherwise.

begin                                                              /* Main */
    $blocks \leftarrow$ all blocks headed by any $B \in \mathsf{S}$ ;
    Mark all blocks $(B, C) \in blocks$ as satisfied where $C = \emptyset$ ;
    repeat
        foreach $(B, C) \in blocks$ that is not marked as satisfied do
            /* Check if there exists a basis $B'$ of $(B, C)$                 */
            foreach $B' \in \mathsf{S} \setminus \{B\}$ do
                if $B'$ is a basis of $(B, C)$ then
                    Mark $(B, C)$ as satisfied;
        if For some $B \in \mathsf{S}$, all blocks headed by $B$ are marked as satisfied then
            return Accept;
    until no new blocks marked;
    return Reject;
```

A LogCFL upper-bound. For the sake of simplicity, the algorithm presented here uses dynamic programming to establish a PTIME upper-bound. However, it is not difficult to see that the ComNF CTD problem lies in the class LogCFL and is therefore highly parallelizable: Consider the LogCFL algorithm for computing hypertree decompositions presented in Reference [31]. To guess the next separator, we now, roughly speaking, guess some element of S instead of guessing a set of up to k edges. Since S is an input, it is sufficient to guess an index into S. The argument for LogCFL membership of computing hypertree decompositions then also applies to computing ComNF CTDs.

The LogCFL upper-bound also extends to our main tractability result for GHDs. Observe that the set S in Lemma 5.18 can be computed in logarithmic space (for fixed a, c, and k). Hence, the proof of Theorem 5.4 also establishes LogCFL membership of computing GHDs of fixed width assuming the LogBMIP. For FHDs it remains open whether this applies. There, our approach requires the computation of the fractional cover number of sets of vertices (cf. Lemma 6.5). No LogCFL algorithm is known for this task.

ACKNOWLEDGMENTS

We are truly grateful to one of the anonymous referees, whose insightful and exceptionally deep review comments have greatly helped to simplify the presentation and increase readability.

REFERENCES

[1] Christopher R. Aberger, Susan Tu, Kunle Olukotun, and Christopher Ré. 2016. EmptyHeaded: A relational engine for graph processing. In *Proceedings of SIGMOD*. ACM, 431–446.

[2] Christopher R. Aberger, Susan Tu, Kunle Olukotun, and Christopher Ré. 2016. Old techniques for new join algorithms: A case study in RDF processing. *CoRR* abs/1602.03557 (2016).

[3] Isolde Adler. 2004. Marshals, monotone marshals, and hypertree-width. *J. Graph Theor.* 47, 4 (2004), 275–296.

[4] Isolde Adler, Georg Gottlob, and Martin Grohe. 2007. Hypertree width and related hypergraph invariants. *Eur. J. Comb.* 28, 8 (2007), 2167–2181.

[5] Foto N. Afrati, Manas Joglekar, Christopher Ré, Semih Salihoglu, and Jeffrey D. Ullman. 2017. GYM: A multiround join algorithm in MapReduce. In *Proceedings of ICDT*, Vol. 68. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 4:1–4:18.

[6] Molham Aref, Balder ten Cate, Todd J. Green, Benny Kimelfeld, Dan Olteanu, Emir Pasalic, Todd L. Veldhuizen, and Geoffrey Washburn. 2015. Design and implementation of the LogicBlox system. In *Proceedings of SIGMOD*. ACM, 1371–1382.

[7] Patrick Assouad. 1983. Densité et dimension. *Annales de l'Institut Fourier* 33, 3 (1983), 233–282.

[8] Albert Atserias, Martin Grohe, and Dániel Marx. 2013. Size bounds and query plans for relational joins. *SIAM J. Comput.* 42, 4 (2013), 1737–1767.

[9] Giorgio Ausiello. 1999. *Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties*. Springer. Retrieved from http://www.worldcat.org/oclc/249492438.

[10] Nurzhan Bakibayev, Tomás Kociský, Dan Olteanu, and Jakub Závodný. 2013. Aggregation and ordering in factorised databases. *PVLDB* 6, 14 (2013), 1990–2001.

[11] Angela Bonifati, Wim Martens, and Thomas Timm. 2017. An analytical study of large SPARQL query logs. *PVLDB* 11, 2 (2017), 149–161. Retrieved from http://www.vldb.org/pvldb/vol11/p149-bonifati.pdf.

[12] H. Brönnimann and M. T. Goodrich. 1995. Almost optimal set covers in finite VC-dimension. *Discr. Comput. Geom.* 14, 4 (01 Dec. 1995), 463–479. DOI: https://doi.org/10.1007/BF02570718

[13] Ashok K. Chandra and Philip M. Merlin. 1977. Optimal implementation of conjunctive queries in relational data bases. In *Proceedings of STOC*. ACM, 77–90.

[14] Chandra Chekuri and Anand Rajaraman. 2000. Conjunctive query containment revisited. *Theor. Comput. Sci.* 239, 2 (2000), 211–229.

[15] Hubie Chen and Víctor Dalmau. 2005. Beyond hypertree width: Decomposition methods without decompositions. In *Proceedings of CP (Lecture Notes in Computer Science)*, Vol. 3709. Springer, 167–181.

[16] Fan R. K. Chung, Zoltan Fueredi, M. R. Garey, and Ronald L. Graham. 1988. On the fractional covering number of hypergraphs. *SIAM J. Discr. Math.* 1, 1 (1988), 45–49.

[17] David A. Cohen, Peter Jeavons, and Marc Gyssens. 2008. A unified theory of structural tractability for constraint satisfaction problems. *J. Comput. Syst. Sci.* 74, 5 (2008), 721–743. DOI: https://doi.org/10.1016/j.jcss.2007.08.001

[18] Víctor Dalmau, Phokion G. Kolaitis, and Moshe Y. Vardi. 2002. Constraint satisfaction, bounded treewidth, and finite-variable logics. In *Proceedings of CP (Lecture Notes in Computer Science)*, Vol. 2470. Springer, 310–326.

[19] Rina Dechter and Judea Pearl. 1989. Tree clustering for constraint networks. *Artif. Intell.* 38, 3 (1989), 353–366.

[20] Guo-Li Ding, Paul Seymour, and Peter Winkler. 1994. Bounding the vertex cover number of a hypergraph. *Combinatorica* 14, 1 (1994), 23–34.

[21] Pierre Duchet. 1996. Hypergraphs. In *Handbook of Combinatorics (Vol. 1)*. The MIT Press, 381–432.

[22] Ronald Fagin. 1983. Degrees of acyclicity for hypergraphs and relational database schemes. *J. ACM* 30, 3 (1983), 514–550.

[23] Wolfgang Fischl, Georg Gottlob, Davide M. Longo, and Reinhard Pichler. 2019. HyperBench: A benchmark and tool for hypergraphs and empirical findings. In *Proceedings of PODS*. ACM, 464–480.

[24] Wolfgang Fischl, Georg Gottlob, and Reinhard Pichler. 2018. General and fractional hypertree decompositions: Hard and easy cases. In *Proceedings of PODS*. ACM, 17–32. DOI:https://doi.org/10.1145/3196959.3196962.

[25] Eugene C. Freuder. 1990. Complexity of k-tree structured constraint satisfaction problems. In *Proceedings of AAAI*. AAAI Press/The MIT Press, 4–9.

[26] Zoltán Füredi. 1988. Matchings and covers in hypergraphs. *Graphs Combinat.* 4, 1 (1988), 115–206.

[27] Nathan Goodman and Oded Shmueli. 1984. The tree projection theorem and relational query processing. *J. Comput. Syst. Sci.* 28, 1 (1984), 60–79. DOI: https://doi.org/10.1016/0022-0000(84)90076-X

[28] Georg Gottlob and Gianluigi Greco. 2013. Decomposing combinatorial auctions and set packing problems. *J. ACM* 60, 4 (2013), 24.

[29] Georg Gottlob, Gianluigi Greco, and Francesco Scarcello. 2018. Tree projections and constraint optimization problems: Fixed-parameter tractability and parallel algorithms. *J. Comput. Syst. Sci.* 94 (2018), 11–40. DOI: https://doi.org/10.1016/j.jcss.2017.11.005

[30] Georg Gottlob, Matthias Lanzinger, Reinhard Pichler, and Igor Razgon. 2020. Fractional covers of hypergraphs with bounded multi-intersection. In *Proceedings of MFCS (Lecture Notes in Computer Science)*, Vol. 170. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 41:1–41:14.

[31] Georg Gottlob, Nicola Leone, and Francesco Scarcello. 2002. Hypertree decompositions and tractable queries. *J. Comput. Syst. Sci.* 64, 3 (2002), 579–627. DOI: https://doi.org/10.1006/jcss.2001.1809

[32] Georg Gottlob, Zoltán Miklós, and Thomas Schwentick. 2009. Generalized hypertree decompositions: NP-hardness and tractable variants. *J. ACM* 56, 6 (2009), 30:1–30:32. DOI: https://doi.org/10.1145/1568318.1568320

[33] Martin Grohe. 2007. The complexity of homomorphism and constraint satisfaction problems seen from the other side. *J. ACM* 54, 1 (2007), 1:1–1:24. DOI: https://doi.org/10.1145/1206035.1206036

[34] Martin Grohe and Dániel Marx. 2006. Constraint solving via fractional edge covers. In *Proceedings of SODA*. ACM Press, 289–298.

[35] Martin Grohe and Dániel Marx. 2014. Constraint solving via fractional edge covers. *ACM Trans. Algor* 11, 1 (2014), 4:1–4:20.

[36] Martin Grohe, Thomas Schwentick, and Luc Segoufin. 2001. When is the evaluation of conjunctive queries tractable? In *Proceedings of STOC*. ACM, 657–666.

[37] Marc Gyssens, Peter Jeavons, and David A. Cohen. 1994. Decomposing constraint satisfaction problems using database techniques. *Artif. Intell.* 66, 1 (1994), 57–89.

[38] Marc Gyssens and Jan Paredaens. 1984. A decomposition methodology for cyclic databases. In *Advances in Data Base Theory: Volume 2*. Springer, 85–122.

[39] Khayyam Hashmi, Zaki Malik, Erfan Najmi, and Abdelmounaam Rezgui. 2016. SNRNeg: A social network enabled negotiation service. *Inf. Sci.* 349 (2016), 248–262.

[40] Mahmoud Abo Khamis, Hung Q. Ngo, Christopher Ré, and Atri Rudra. 2015. Joins via geometric resolutions: Worst-case and beyond. In *Proceedings of PODS*. ACM, 213–228.

[41] Mahmoud Abo Khamis, Hung Q. Ngo, and Atri Rudra. 2016. FAQ: Questions asked frequently. In *Proceedings of PODS*. ACM, 13–28.

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

[43] Aviv Lustig and Oded Shmueli. 1999. Acyclic hypergraph projections. *J. Algor.* 30, 2 (1999), 400–422. DOI: https://doi.org/10.1006/jagm.1998.0965

[44] Dániel Marx. 2010. Approximating fractional hypertree width. *ACM Trans. Algor.* 6, 2 (2010), 29:1–29:17. DOI: https://doi.org/10.1145/1721837.1721845

[45] Dániel Marx. 2011. Tractable structures for constraint satisfaction with truth tables. *Theor. Comput. Syst.* 48, 3 (2011), 444–464.

[46] Dániel Marx. 2013. Tractable hypergraph properties for constraint satisfaction and conjunctive queries. *J. ACM* 60, 6 (2013), 42:1–42:51.

[47] Lukas Moll, Siamak Tazari, and Marc Thurley. 2012. Computing hypergraph width measures exactly. *Inf. Process. Lett.* 112, 6 (2012), 238–242. DOI: https://doi.org/10.1016/j.ipl.2011.12.002

[48] Dan Olteanu and Jakub Závodný. 2015. Size bounds for factorised representations of query results. *ACM Trans. Datab. Syst.* 40, 1 (2015), 2:1–2:44. DOI: https://doi.org/10.1145/2656335

[49] Norbert Sauer. 1972. On the density of families of sets. *J. Combinat. Theor. (A)* 13, 1 (1972), 145–147.

[50] Susan Tu and Christopher Ré. 2015. DunceCap: Query plans using generalized hypertree decompositions. In *Proceedings of SIGMOD*. ACM, 2077–2078.

[51] René van Bevern, Rodney G. Downey, Michael R. Fellows, Serge Gaspers, and Frances A. Rosamond. 2015. Myhill-Nerode methods for hypergraphs. *Algorithmica* 73, 4 (2015), 696–729. DOI: https://doi.org/10.1007/s00453-015-9977-x

[52] Vladimir Vapnik and Alexey Chervonenkis. 1971. On the uniform convergence of relative frequencies of events to their probabilities. *Theor. Probab. Appl.* 16 (1971), 264–280.

[53] Vijay V. Vazirani. 2001. *Approximation Algorithms*. Springer. Retrieved from http://www.springer.com/computer/theoretical+computer+science/book/978-3-540-65367-7.

[54] Mihalis Yannakakis. 1981. Algorithms for acyclic database schemes. In *Proceedings of VLDB*. IEEE Computer Society, 82–94.

Received November 2018; revised January 2021; accepted March 2021