More consequences of falsifying SETH and the orthogonal vectors conjecture

More Consequences of Falsifying SETH and the Orthogonal Vectors Conjecture

Amir Abboud
IBM Almaden Research Center, San Jose, CA, USA
amir.abboud@ibm.com

Karl Bringmann
Max Planck Institute for Informatics, Saarland Informatics Campus, Saarbrücken, Germany
kbringma@mpi-inf.mpg.de

Holger Dell
Saarland University and Cluster of Excellence (MMCI), Saarbrücken, Germany
hdell@mmci.uni-saarland.de
https://orcid.org/0000-0001-8955-0786

Jesper Nederlof
Eindhoven University of Technology, Eindhoven, The Netherlands
j.nederlof@tue.nl

Abstract

The Strong Exponential Time Hypothesis and the OV-conjecture are two popular hardness assumptions used to prove a plethora of lower bounds, especially in the realm of polynomial-time algorithms. The OV-conjecture in moderate dimension states there is no $\varepsilon > 0$ for which an $O(N^{2-\varepsilon})$ $\mathrm{poly}(D)$ time algorithm can decide whether there is a pair of orthogonal vectors in a given set of size N that contains D-dimensional binary vectors.

We strengthen the evidence for these hardness assumptions. In particular, we show that if the OV-conjecture fails, then two problems for which we are far from obtaining even tiny improvements over exhaustive search would have surprisingly fast algorithms. If the OV conjecture is false, then there is a fixed $\varepsilon > 0$ such that:

1. For all d and all large enough k, there is a randomized algorithm that takes $O(n^{(1-\varepsilon)k})$ time to solve the Zero-Weight-k-Clique and Min-Weight-k-Clique problems on d-hypergraphs with n vertices. As a consequence, the OV-conjecture is implied by the Weighted Clique conjecture.
2. For all c, the satisfiability of sparse TC^1 circuits on n inputs (that is, circuits with cn wires, depth $c \log n$, and negation, AND, OR, and threshold gates) can be computed in time $O((2-\varepsilon)^n)$.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases fine-grained complexity, OV, clique, satisfiability, threshold circuits

Related Version Proceedings version doi:10.1145/3188745.3188938 to appear at the 50th Annual ACM SIGACT Symposium on the Theory of Computing, June 25–29, 2018, Los Angeles, CA, USA.

1 Introduction

The Strong Exponential Time Hypothesis (SETH) is a cornerstone of contemporary algorithm design that was formulated by Impagliazzo and Paturi [34] and recently gained extensive popularity. It postulates that exhaustive search is essentially the fastest possible method to

decide the satisfiability of bounded-width CNF formulas. SETH is used in the study of exact and fixed parameter tractable algorithms, see e.g [23, 46] or the book by Cygan et al. [24]. In this area, it implies, among other things, tight lower bounds for problems on graphs that have small treewidth or pathwidth [41, 26, 25].

Closely related to SETH, the orthogonal vectors problem (OV) is, given two sets A and B of N vectors from $\{0,1\}^D$, to decide whether there are vectors $a \in A$ and $b \in B$ such that a and b are orthogonal in $\mathbf{Z}^D$. If $D \leq O(N^{0.3})$ holds, the problem can be solved in time $\tilde{O}(N^2)$ using an algorithm based on fast rectangular matrix multiplication (see e.g. [31]). SETH implies [54] that this algorithm is essentially as fast as possible; in particular, SETH implies the following hardness conjecture, which was given its name by Gao et al. [32].

▶ Conjecture 1.1 (Moderate-dimension OV Conjecture). There are no reals $\varepsilon, \delta > 0$ such that OV for $D = N^\delta$ can be solved[1] in time $O(N^{2-\varepsilon})$.

The moderate-dimension OV conjecture is used to study the fine-grained complexity of problems in P, for which it has remarkably strong and diverse implications. If the conjecture is true, then dozens of important problems from all across computer science exhibit running time lower bounds that match existing upper bounds up to subpolynomial factors. These include pattern matching and other problems in bioinformatics [7, 10, 40, 1], graph algorithms [47, 6, 32], computational geometry [16], formal languages [11, 18], time-series analysis [2, 19], and even economics [42] (see [58] for a more comprehensive list).

Gao et al. [32] also named the *low-dimension OV conjecture*, which asserts that OV does not have subquadratic algorithms whenever $D = \omega(\log N)$ holds. The low-dimension implies the moderate-dimension variant of the OV conjecture, and both are implied by SETH [54]. Recent results on the hardness of approximation problems, such as Maximum Inner Product [5], rely on the stronger conjecture (perhaps also [12, 14]). However, for the vast majority of OV-based hardness results, reducing the dimension only affects lower-order terms in the lower bounds and so it often suffices to assume the moderate-dimension variant. Doing so makes results stronger, and it is this variant of the OV conjecture that we strengthen further in the present work.

1.1 Other conjectures

Two other popular conjectures in fine-grained complexity make assertions for the All-Pairs-Shortest-Path Problem (APSP)[2] and the 3-SUM problem[3]. It is an important and long-standing open question to determine the relationship between APSP, 3-SUM, and OV. In particular, it is open whether the APSP conjecture or the 3-SUM conjecture imply the moderate-dimension OV conjecture.

Closely related to APSP is the Min-Weight-k-Clique problem: Given a graph on n nodes with integer edge-weights in some polynomial range, the goal is to find a k-clique of minimum weight. The exhaustive search algorithm solves this problem in $O(n^k)$ time, and for $k = 3$,

[1] In this work we hardly distinguish between randomized and deterministic algorithms, as even randomized algorithms with the desired running times would constitute an important breakthrough.

[2] The APSP problem is to compute all pairwise distances in a graph (given by its adjacency matrix) on n nodes and with edges weights in some polynomial range. It is conjectured to require $n^{3-o(1)}$ time, and many problems, especially on graphs, are known to be equivalent to or at least as hard as APSP, see e.g. [48, 60, 6, 3, 49, 8, 27].

[3] The 3-SUM problem is to decide if a given set of n integers contains three that sum to zero. It is conjectured that the problem requires $n^{2-o(1)}$ time. Many problems, especially in computational geometry, are known to be 3-SUM-hard, see [30].

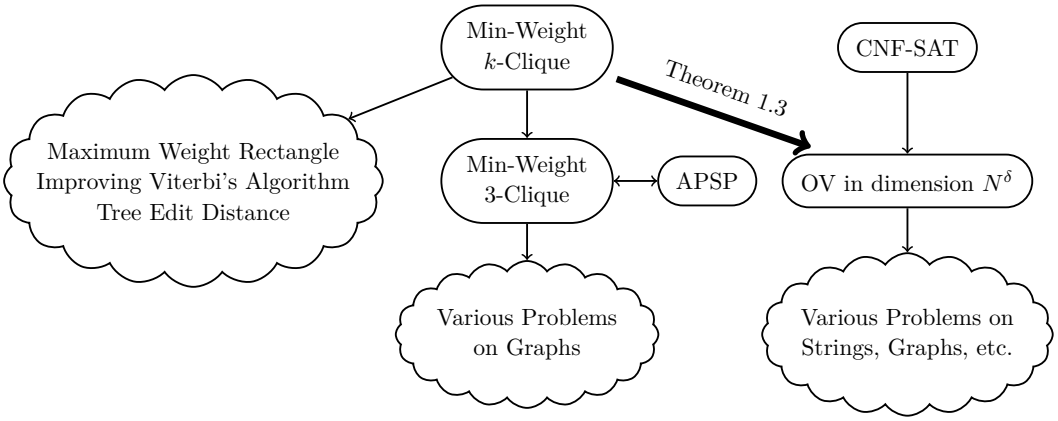

Figure 1 Illustration of the landscape of Hardness in P. An arrow from problem A to problem B indicates that improving the runtime of problem B from T_B to $T_B^{1-\varepsilon}$ implies an improvement for problem A from T_A to $T_A^{1-\varepsilon'}$. Our contribution is the bold black arrow (Theorem 1.3).

the problem is subcubically-equivalent to the APSP conjecture [59]. That is, either both APSP and Min-Weight-3-Clique have algorithms running in time $O(n^{3-\varepsilon})$ for some $\varepsilon > 0$, or neither of them do.

For all integers $k \geq 3$, there is a simple reduction from Min-Weight-k-Clique to Min-Weight-3-Clique, and by combining it with the fastest known APSP algorithm [56, 21], we can solve Min-Weight-k-Clique in time $n^k / \exp\big(\Omega(\sqrt{\log n})\big)$. The improvement of this algorithm over exhaustive search is subpolynomial. Due to the equivalence with APSP for $k = 3$, it is natural to conjecture that a truly polynomial advantage in the running time is impossible.

▶ Conjecture 1.2 (Weighted Clique Conjecture). There is no real $\varepsilon > 0$ and integer $k \geq 3$ such that the Min-Weight-k-Clique problem on n-vertex graphs and with edge-weights in $\{-M, \ldots, M\}$ can be solved in time $O(n^{(1-\varepsilon)k}) \operatorname{polylog} M$.

This conjecture implies the APSP conjecture, and thus implies lower bounds for all problems that are known to be APSP-hard [48, 60, 6, 3, 49, 8, 27]. In addition, the Weighted Clique conjecture implies lower bounds for a variety of problems that are not known to be APSP-hard: the Local Alignment problem [7] from bioinformatics, the Maximum Rectangle problem [9] from computational geometry, the Viterbi problem [13] from machine learning and, the Tree Edit Distance problem [17].

1.2 Our Results for OV

We prove that the Weighted Clique conjecture (Conjecture 1.2) implies the moderate-dimension OV conjecture (Conjecture 1.1). To this end, we design a tight randomized reduction from Min-Weight-k-Clique to OV. The impact of this result on fine-grained complexity in P is depicted in Figure 1. As can be seen, we identify Min-Weight-k-Clique as a core problem in this landscape, since it tightly reduces to most problems that have known conditional lower bounds; the main exceptions are 3-SUM-hard problems and the few problems that do require the low-dimension version of the OV conjecture.

In fact, our result is even stronger: We show that improved algorithms for moderate-dimension OV leads to improved algorithms for finding weighted cliques *even in hypergraphs*. A d-hypergraph is a hypergraph in which all edges are of size at most d. A *clique* of a d-hypergraph G is a subset $X \subseteq V(G)$ such that for every $e \subseteq X$ of size at most d we

have $e \in E(G)$. The *k-Clique* problem is given G as input to find such a clique X of size k. We also study weighted versions, where we are additionally given an edge-weight function $w : E(G) \to \mathbf{Z}$ and a target integer $t \in \mathbf{Z}$. The weight of a clique X in G is the sum $\sum_e w(e)$ over all edges $e \in E(G)$ with $e \subseteq X$. In the *Exact-Weight-k-Clique* problem, we need to find a k-clique X of weight exactly t, and in *Min-Weight-k-Clique* we need one of weight at most t. We are ready to formally state our first theorem.

▶ **Theorem 1.3.** *If the moderate-dimension OV conjecture is false, then there exists an $\varepsilon > 0$ such that for every integer d there is a (large) integer $k = k(d, \varepsilon)$ satisfying the following statements:*

- *k-Clique can be solved on d-hypergraphs in time $O(n^{(1-\varepsilon)k})$.*
- *Exact-Weight-k-Clique and Min-Weight-k-Clique can be solved on d-hypergraphs with weights in $\{-M, \ldots, M\}$ in randomized time $O(n^{(1-\varepsilon)k}) \cdot \operatorname{polylog} M$.*

Clique problems on hypergraphs appear to be harder than on graphs. For example, in the unweighted case, k-Clique on graphs can be solved in $O(n^{0.79k})$ time [44, 29], whereas on 3-hypergraphs no polynomial improvement over the $O(n^k)$ exhaustive search algorithm is known. If the moderate-dimension OV conjecture is false, then Theorem 1.3 implies that k-Clique on 3-hypergraphs does have such an improvement for some large enough integer k. This strengthens the OV conjecture.

Significantly improved algorithms for k-Clique on d-hypergraphs are not only unknown, they are in fact known to imply breakthrough algorithms for Max-d-SAT, the optimization version of the d-CNF-SAT problem that needs to find an assignment that satisfies as many clauses as possible. Using this known reduction, we obtain the following corollary to Theorem 1.3.

▶ **Corollary 1.4.** *If the moderate-dimension OV conjecture is false, then there exists an $\varepsilon > 0$ such that, for all integers d, there is an algorithm for Max-d-SAT that runs in time $O^*\big((2-\varepsilon)^n\big)$.*

Due to this corollary, the moderate-dimension OV conjecture is not only implied by SETH, but even by its stronger cousin for the Max-SAT problem on bounded-width CNF. As a consequence, all known lower bounds based on the moderate-dimension OV conjecture are now automatically based on the hardness of bounded-width Max-SAT rather than just bounded-width CNF-SAT. Corollary 1.4 subsumes some results [2, 8, 39] where this was done in special cases.

The implications of Theorem 1.3 for the weighted problems strengthen the moderate-dimension OV conjecture further. Any algorithm that solves Exact-Weight-k-Clique can in particular solve Zero-Weight-k-Clique where the target weight satisfies $t = 0$. In turn, any algorithm for the latter problem can also be used to solve Min-Weight-k-Clique without significant running time overhead [43]. While the best known algorithms for Min-Weight-k-Clique run in time $n^k / \exp\big(\Omega(\sqrt{\log n})\big)$ [56, 21], such superpolylogarithmic shavings are open for Zero-Weight-k-Clique. The $k = 3$ case is particularly interesting, since solving Zero-Weight-3-Clique in $O(n^{3-\varepsilon})$ time refutes not only the APSP conjecture but *also* the 3-SUM conjecture [60, 45, 38].

Proof ideas

We prove Theorem 1.3 by designing a tight reduction from Min-Weight-k-Clique on d-hypergraphs to OV. We sketch the reduction for $d = 2$. It has two main *stages*. In the first,

we reduce Min-Weight-k-Clique on graphs to unweighted k-Clique on 4-hypergraphs, where each hyperedge has cardinality at most 4. We achieve this in a sequence of weight reduction *steps*: We start with a standard hashing trick to reduce the weights to a polynomial range. Then, to reduce the weights further, we chop the bits of the numbers into vectors and use a squaring trick to combine all the coordinates. This trick is borrowed from [4], where it was used to reduce node weights in graphs. We instead use it to reduce edge weights, which however we only achieve by transforming the graph into a 4-hypergraph. Finally, once the weights are small enough, we remove them completely via an exhaustive search.

In the second stage, we reduce the unweighted k-Clique problem on 4-hypergraphs to a k-wise variant of OV. The reduction maps each node to a Boolean vector by encoding the incident hyperedges into the coordinates such that a disjointness check among k vectors corresponds to checking that k nodes form a hyperclique. Finally, we reduce the k-wise variant to the OV problem using a standard reduction.

1.3 Our results for SETH and CNF-SAT

There are many algorithms that solve d-CNF-SAT, the satisfiability problem on d-CNF formulas, in time $O^*\big((2-\varepsilon_d)^n\big)$. As d grows to infinity, the constants ε_d for all these algorithms tend to 0 in the limit. SETH was conceived by this observation, and it asserts exactly this: If ε_d is the largest real such that d-CNF-SAT can be solved in time $O^*((2-\varepsilon_d)^n)$, then $\lim_{d\to\infty} \varepsilon_d = 0$ holds. Thus SETH is not about the hardness of an individual problem, but about a sequence of problems each of which we know to have a faster algorithm than exhaustive search. This makes it easier prove lower bounds under SETH, but it is unfortunate if we want to have confidence that SETH and the implied lower bounds are true.

Indeed, there are algorithms that get substantial $n^{\omega(1)}$ speed-ups over 2^n for CNF formulas of unbounded width (see e.g. [15, 20]). If instead of CNF formulas, we consider more complex Boolean circuits, such as bounded-depth threshold circuits (TC^0-circuits), then we should get a computationally harder satisfiability problem. For linear-size threshold circuits of depth two, there are satisfiability algorithms that run in time $O^*\big((2-\varepsilon_c)^n\big)$; here too, the constants ε_c tend to 0 as c grows [35, 22]. However, even for linear-size threshold circuits of depth 4, satisfiability algorithms with speed-up $n^{\omega(1)}$ are unknown. Obtaining such algorithms for linear-size TC^0 would resolve Williams' question [57] of whether his circuit lower bound framework can prove $\mathrm{NEXP} \not\subseteq \mathrm{TC}^0$. We show that a refutation of SETH would constitute progress on these questions.

▶ **Theorem 1.5.** *If SETH fails, then there is an $\varepsilon > 0$ such that, for all constants c and d, the satisfiability of depth-d threshold circuits with cn wires can be determined in time $O^*\big((2-\varepsilon)^n\big)$.*

This theorem is the newest member in a sequence of increasingly general results of Santhanam and Srinivasan [50], Dantsin and Wolpert [28], and Cygan et al. [23], who show that refuting SETH implies faster algorithms for the satisfiability of linear-size formulas, linear-size AC^0-circuits, and linear-size VSP-circuits, respectively. Our class of linear-size TC^0-circuits contains the classes of linear-size formulas and AC^0-circuits, so we generalize these two results. The class VSP is less understood and little is known about its complexity properties. While algorithms with running time $2^n/n^{\omega(1)}$ are known for the satisfiability of linear-size AC^0-circuits [33], such algorithms are not known for linear-size TC^0-circuits, even when the depth is 4 and the number of wires is $10n$.

As usual with reductions, a pious believer is biased to view Theorem 1.5 as another confirmation that SETH and all its logical implications are indeed true, which includes the

moderate-dimension OV conjecture. On the other hand, a skeptic who is hesitant to believe SETH or one of its implications is now invited to start their refutation attempt by providing faster algorithms for linear-size TC^0-circuits, since any refutation of SETH would have to do that implicitly.

Extension for CNF-SAT

Much like most OV-based lower bounds in P can be based on the moderate-dimension OV conjecture rather than its low-dimension variant, many SETH-based lower bounds for exponential time and parameterized problems can be based on the weaker assumption that satisfiability cannot be solved in time $O^*\big((2-\varepsilon)^n\big)$ for CNF formulas of unbounded width. This weaker assumption for CNF-SAT suffices, for example, in results for graph problems that have small treewidth or pathwidth [41, 26, 25]. We add further weight to these hardness results by showing that sufficiently fast algorithms for CNF-SAT imply improved satisfiability algorithms for linear-size threshold circuits of super-logarithmic depth, which is a larger class than TC^1-circuits.

▶ **Theorem 1.6.** *If CNF-SAT can be solved in $O^*(2^{(1-\varepsilon)n})$ time for some $\varepsilon > 0$, then there is an $\varepsilon' > 0$ such that, for all $c > 0$, there is a $\delta > 0$ such that the satisfiability for threshold circuits of depth $(\log n)^{1+\delta}$ and at most cn wires can be determined in time $O(2^{(1-\varepsilon')n})$.*

If SETH is false, then not every problem in E^{NP} can be computed by linear-size VSP-circuits [55, 36]. By Theorem 1.6, solving CNF-SAT in time $O^*\big((2-\varepsilon)^n\big)$ implies the perhaps more natural result that not every problem in E^{NP} has linear-size TC^1-circuits.

Proof idea

Similar to the analogous result by Cygan et al. [23] for VSP-circuits, we use a depth reduction technique introduced by Valiant [52], which shows that VSP-circuits embed nicely into CNF-formulas. We use an additional trick that allows us to get rid of threshold gates.

2 Preliminaries

Notation

The $O^*(\cdot)$ and $\tilde{O}(\cdot)$ notations omit factors that are polynomial and polylogarithmic in the input size, respectively. We write $\mathbf{Z}$ for the integers and $\mathbf{N}$ for the non-negative integers. We let $[n] = \{1, \ldots, n\}$ for $n \in \mathbf{N}$. If S is a set, we write $\binom{S}{d}$ for the set of all subsets of S that have size exactly d, and $\binom{S}{\leq d}$ for the set of all subsets of size at most d. A d-hypergraph G for $d \in \mathbf{N}$ is a tuple $(V(G), E(G))$, where $V(G)$ is a finite set of *vertices* and $E(G) \subseteq \binom{V(G)}{\leq d}$ is a set of *edges*. If G is a d-hypergraph and $X \subseteq V(G)$, then $G[X]$ denotes the *subgraph induced by* X, that is, $V(G[X]) = X$ and $E(G[X]) = E(G) \cap \binom{X}{\leq d}$. A set $S \subseteq V(G)$ is called a clique in G if $E(G[S]) = \binom{S}{\leq d}$. A k-clique is a clique of size k.

A *graph* is a 2-hypergraph. In contrast to usual graph notation,[4] there are also edges $\{v\}$ of size 1 and the edge $\emptyset$ of size 0; a k-clique is a set $\{v_1, \ldots, v_k\}$ for which every pair $\{v_i, v_j\}$

[4] We remark that there is an alternative definition of hypergraphs and cliques, where each edge of a d-hypergraph has size exactly d instead of at most d, and a clique is a set S such that $E(G[S]) = \binom{S}{d}$. The two variants are equivalent in terms of the algorithmic problem of deciding whether G contains a k-clique. Indeed, if we want to detect a set S with $E(G[S]) = \binom{S}{d}$, then we can add all sets of size at most $d-1$ to $E(G)$ and then detect a set S with $E(G[S]) = \binom{S}{\leq d}$. Similarly, if we want to detect a set

is in $E(G)$, every singleton $\{v_i\}$ is in $E(G)$, and $\emptyset \in E(G)$. This does not significantly change the problem of detecting whether G contains a k-clique, since testing whether $\emptyset \in E(G)$ is in constant time, and we can assume without loss of generality that $\{v\} \in E(G)$ for all $v \in V(G)$, by deleting all other vertices.

CNF-SAT

The d-SAT problem is to determine whether a given d-CNF formula has a satisfying assignment. We denote the number of variables by n and define s_d as the real number

$$\inf\{\delta > 0 : \text{there is an } O(2^{\delta n}) \text{ time algorithm for } d\text{-SAT}\}.$$

Let $s_\infty = \lim_{d\to\infty} s_d$. Impagliazzo and Paturi's *Strong Exponential Time Hypothesis (SETH)* postulates that $s_\infty = 1$ holds [34].

DAGs and Circuits

If G is a directed acyclic graph (DAG), we let $N_G^-(v)$ denote the set of in-neighbors of v and let $d_G^-(v)$ denote the *in-degree* with $d_G^-(v) = |N_G^-(v)|$. The *depth* of G is the length of the longest directed path in it.

A *Boolean function* is any function $f : \{0,1\}^d \to \{0,1\}$. It is *symmetric* if $f(x) = f(y)$ holds for all $x, y \in \{0,1\}^d$ whose Hamming weight is the same. Let B be a set of symmetric Boolean functions. A *(Boolean) circuit C over a basis B* is a pair (G, λ) where G is a directed acyclic graph and $\lambda \in B^V$ is a labeling of its vertex set V with elements from B. We say that v is a *λ_v-gate*, and we require that the in-degree of v is equal to the arity of λ_v, that is, we have $\lambda_v : \{0,1\}^{d_G^-(v)} \to \{0,1\}$. The edges of G are called *wires*, the in-degree of a gate is called its *fan-in*, and we write $V(C)$ for $V(G)$. The set of *input gates* $I(C)$ or $I(G)$ of C consists of the vertices with in-degree 0, and the set of *output gates* $O(C)$ or $O(G)$ of C consists of the vertices with out-degree 0. If $x \in \{0,1\}^{I(G)}$ is a setting for the input gates, we define $C_v(x)$ as the *value of C at $v \in V$ on input x* inductively: If $v \in I(C)$, let $C_v(x) = x_v$, and otherwise, let $C_v(x) = \lambda_v(C_{v_1}(x), \ldots, C_{v_\ell}(x))$, where $v_1, \ldots, v_\ell$ denotes the in-neighbors of v in G; note that this is well-defined since G is acyclic and λ_v is symmetric. Slightly abusing notation, we may write C also for the function $C : \{0,1\}^{I(G)} \to \{0,1\}^{O(G)}$ with $C(x) = (C_v)_{v \in O(G)}$. Or we may view circuits as mapping integers to integers in a fixed range $[r]$ for convenience while in fact this is implemented by storing the binary representation of these values with $\lceil \lg r \rceil$ gates.

A (u,v)-path in C is a directed path $u_1, \ldots, u_\ell$ in G with $u = u_1$ and $u_\ell = v$. If $A \subseteq V(C)$, we let $R_C(A, v)$ denote the set of vertices from which v is reachable without using vertices of A, that is,

$$R_C(A, v) = \{u \in V : \ G[V \setminus A \cup \{u, v\}] \text{ contains } (u,v)\text{-path } \}. \tag{1}$$

Finally, for a circuit C, a gate $v \in V(C)$, and a set $A \subseteq V(C)$, we define $C_{v,A}$ as the subcircuit of C that is induced by the set $R_C(A, v)$; note that v is the only output gate of $C_{v,A}$ and its input gates are contained in $A \cup I(C)$.

S with $E(G[S]) = \binom{S}{\leq d}$, then we can build a new hypergraph $G' = (V(G), E')$ where E' contains all sets $e \subseteq V(G)$ of size d whose every subset is in $E(G)$, and then detect a set S with $E(G'[S]) = \binom{S}{d}$. Similar equivalences hold for the weighted variants of the k-Clique problem. Thus, our choice of a variant is only for notational convenience.

We use the Boolean functions $\mathrm{NEG}(x) = \neg x$, $\mathrm{AND}(x,y) = x \wedge y$, $\mathrm{OR}(x,y) = x \vee y$ and $\mathrm{TH}_\theta : \{0,1\}^d \to \{0,1\}$ which is, for every positive $\theta \leq n$ defined to be 1 if $\sum_{i=1}^{d} x_i \geq \theta$ and to be 0 otherwise. Note that $\mathrm{AND}(x,y) = \mathrm{TH}_2(x,y)$ and $\mathrm{OR}(x,y) = \mathrm{TH}_1(x,y)$. We also use $\mathrm{MOD}_m(x_1,\ldots,x_d)$ for $m \leq d$ which is defined to be 1 if m divides $\sum_{i=1}^{d} x_i$ and to be 0 otherwise, and $\mathrm{MAJ}(x_1,\ldots,x_d) = \mathrm{TH}_{d/2}(x_1,\ldots,x_d)$.

A Boolean circuit over the basis $\{\mathrm{NEG}, \mathrm{AND}, \mathrm{OR}, \mathrm{TH}_\theta\}$, where all gates (except for NEG) may have unbounded fan-in, is called a *threshold circuit* (TC); we use AND and OR only for syntactic convenience as they can be simulated by TH_θ. The problem TC-SAT is, given a threshold circuit C with exactly one output gate, to decide whether the circuit is satisfiable, that is, whether there exists a setting $x \in \{0,1\}^n$ for the n input gates such that $C(x) = 1$. For $d \in \mathbf{N}$ and $c > 0$, a *c-sparse-d-depth-TC* is a threshold circuit with n variables, at most cn wires, and depth at most d. For each $i \in \mathbf{N}$, a TC^i-circuit is a family of threshold circuits of depth $O(\log^i n)$ and size $\mathrm{poly}(n)$.

3 Weighted Cliques in Hypergraphs

Recall that in the Exact-Weight-k-Clique problem on d-hypergraphs we are given a d-hypergraph G and a target value t, and the task is to decide whether some size-k subset $S \subseteq V(G)$ forms a clique of total weight $\sum_{e \in E(G[S])} w(e) = t$. We denote by $M = M(w,t)$ the maximum weight in absolute value, that is, we have $M = \max(\{|t|\} \cup \{|w(e)| : e \in E(G)\})$. We write $n = |V(G)|$. Since in this section we will mostly deal with the Exact-Weight-k-Clique on d-hypergraphs problem, we will abbreviate it to "Weighted d-Hypergraph k-Clique".

3.1 Preprocessing Reductions

We rely on some basic reductions: The first turns the hypergraph into a complete d-hypergraph, which shows that the graph structure is immaterial for this problem; the second makes the hypergraph k-partite, which will be useful in our constructions; the third reduces from "exact weight clique" to "zero weight clique", that is, it sets the target value t to 0 by using negative edge weights; the fourth uses a non-negative target value but removes negative weights. In the following statement, M' denotes the maximum weight $M(w',t')$ of the respective output instance.

▶ Fact 3.1. Let $d, k \in \mathbf{N}$ with $1 \leq d \leq k$. There are $O(n^d)$-time self-reductions for Weighted d-Hypergraph k-Clique with the following properties:

1. "Make complete": maps an instance (G,w,k,t) to (G',w',k,t') with $V(G') = V(G)$, $E(G') = \binom{V(G)}{\leq d}$, and $M' \leq \binom{k}{\leq d} M$.
2. "Make k-partite": maps an instance (G,w,k,t) to (G',w',k,t) with $|V(G')| \leq k|V(G)|$ and $M = M'$, such that G' is k-partite in the sense that $V(G')$ is partitioned into k parts and every edge intersects each part in at most one vertex.
3. "Make target zero": maps a k-partite instance (G,w,k,t) to (G,w',k,t') with $t' = 0$ and $M' \leq 2M$.
4. "Make weights non-negative": maps an instance (G,w,k,t) to (G,w',k,t') with $w' : E(G) \to \mathbf{N}$ and $M' \leq 2\binom{k}{\leq d}^2 M$.

Proof. Let (G,w,k,t) be an instance for the problem.

For the first claim, we set $w(e) = \binom{k}{\leq d} M$ for edges e that are supposed to be absent; such edges cannot be used by any solution. Hence, we can assume $E(G) = \binom{V(G)}{\leq d}$ without loss of generality.

For the second claim, we define $V(G') = \{1, \dots, k\} \times V(G)$. For every pairwise distinct $a_1, \dots, a_{d'} \in \{1, \dots, k\}$ and every edge $\{v_1, \dots, v_{d'}\} \in E(G)$ of size d', we add an edge f with

$$f = \big\{(a_1, v_1), \dots, (a_{d'}, v_{d'})\big\} \subseteq V(G')$$

to G'. We set the weight $w'(f) = w(\{v_1, \dots, v_{d'}\})$. It is clear that this instance is equivalent to the input instance, and k-partite (the parts consist of vertices with equal first coordinate).

For the third claim, we slightly modify the weights by setting $t' = 0$ and subtracting t from certain edge weights. Specifically, we start with the construction used in the second claim. For any edge of cardinality d, denoted by $f = \{(a_1, v_1), \dots, (a_d, v_d)\}$, we set $w'(f) = w(\{v_1, \dots, v_d\})$ if $\{a_1, \dots, a_d\} \neq \{1, \dots, d\}$ and $w'(f) = w(\{v_1, \dots, v_d\}) - t$ if $\{a_1, \dots, a_d\} = \{1, \dots, d\}$. Note that any k-clique in G' contains exactly one edge f that intersects the first d parts of the k-partition in exactly one vertex each.

For the fourth claim, we first ensure that $E(G) = \binom{V(G)}{\leq d}$ using the first claim, which increases M by at most a factor $\binom{k}{\leq d}$. Let $L = \max\{0, -w(e) : e \in E(G)\}$, that is, L is the absolute value of the smallest negative weight that occurs in the input, or 0 if there is none. We set $w'(e) = w(e) + L$ for all e and $t' = t + L\binom{k}{\leq d}$. If $t' < 0$ or $t' > \max\{w'(e)\} \cdot \binom{k}{\leq d}$, the instance is a trivial no-instance. Otherwise the reduction outputs (G, w', k, t'). ◀

3.2 Weight Reduction: From Arbitrary to Polynomial

We proceed by reducing the weights of a given instance of the Weighted d-Hypergraph k-Clique problem. By taking the numbers modulo a random prime, we reduce the maximum weight from M to $n^{O(k)}$ in the following way.

▶ **Lemma 3.2.** *Let $d, k \in \mathbf{N}$ with $1 \leq d \leq k$. For some constant $f(k,d) \in \mathbf{N}$ there is a randomized $f(k,d) \cdot \operatorname{polylog} M$ time self-reduction for the Weighted d-Hypergraph k-Clique problem that, on input an instance (G, w, k, t) with maximum weight M, makes at most $f(k,d)$ queries to instances (G, w', k, t') where $w' : E(G) \to \mathbf{N}$, $t' \in \mathbf{N}$, $M' \leq f(k,d) \cdot n^{O(k)}$, and the success probability of the reduction is at least 99%.*

Proof. If $M \leq n^k$ holds, we do not need to do anything. If $M \geq \exp(n^k)$ holds, then in time $O(n^k \log M) = \operatorname{polylog}(M)$ we brute-force the problem. In the remaining case, we sample a prime p uniformly at random from a range specified later, and set $w'(e) = w(e) \bmod p$ for all e. We query the oracle (G, w', k, t') for all t' with $t' = jp + (t \bmod p)$ and $j \in \{0, \dots, \binom{k}{\leq d}\}$, and we output *yes* if and only if at least one oracle query returns *yes*. To prove the completeness of this reduction, let S be a k-clique of weight t with respect to w. Then $\sum_{e \in E(G[S])}(w(e) \bmod p) = jp + \big((\sum_{e \in E(G[S])} w(e)) \bmod p\big) = jp + (t \bmod p) = jp + t'$ holds for some j in the specified range since $\binom{k}{\leq d}$ is the number of terms in the sum. Hence yes-instances map to yes-instances with probability 1. Conversely, if such a j exists, then the weight of S modulo p is equal to t' modulo p.

For the soundness of the reduction, we need to specify the sampling process for p. This is implemented as follows: let $Q = 200n^k \log(k^d M)$ and sample positive integers bounded by $O(Q \ln Q)$ uniformly at random until we have found a prime (which we can verify, for example, deterministically in time $O(\operatorname{polylog} Q) = O(\operatorname{polylog} M)$ since $M > n$). By the prime number theorem, with probability at least 99.5%, after $O(\ln Q) \leq O(dk \log n)$ samples we have found a prime that is a uniform sample from a set of at least Q primes.

The weight of each k-clique S in G is at most $k^d M$ in absolute value, and there are at most n^k distinct sets S, and so n^k is also an upper bound on the number of distinct weights

that appear. For the soundness of the reduction, it is sufficient that $w(S)$ is not congruent to t modulo p. As $|t - w(S)|$ is at most $k^d M$, it has at most $\log(k^d M)$ prime divisors. Therefore the probability that for some S it holds that $w(S)$ is congruent to t modulo p is at most $n^k \log(k^d M)/Q = 1/200$. Overall, we succeed at finding a prime with the desired property with probability $(99.5\%)^2 \geq 99\%$.

We indeed make at most k^d queries to the oracle, and the largest weight in each query is bounded by $\binom{k}{\leq d} \cdot p$. Since p is bounded by Q and $M \leq \exp(n^k)$, this is at most $f(k,d)n^{O(k)}$. For the running time, we need to worry about the bitlength of the involved weights. The input weights use at most $\log M$ bits, and so Q (and thus any p) uses at most $O(dk \log n + \log M) = \operatorname{polylog} M$ bits. Computing the weights modulo p can be done in time $\operatorname{polylog} M$. ◀

3.3 Weight Reduction: From Polynomial to Unweighted

We reduce the weights from $n^{O(k)}$ to $f(k,d) \cdot \log n$ using a deterministic argument, and then use exhaustive search to reduce to the unweighted case.

The *q-expansion* of a number $N \in \mathbf{N}$ is the unique sequence $N_0, N_1, \cdots \in \{0, \ldots, q-1\}$ with $N = \sum_{\ell \in \mathbf{N}} N_\ell q^\ell$. After applying the q-expansion, all N_ℓ are bounded by $q-1$. However, the smaller we choose q, the longer the relevant part of the encoding of N gets; the precise length of this encoding is $p = \lceil \log_q(N+1) \rceil$. The following lemma uses carries to split the weight constraint along the q-expansions of the edge weights w and the target t.

▶ **Lemma 3.3.** *Let G be a d-hypergraph with edge-weight function $w : E(G) \to \mathbf{N}$ and a set $S \subseteq V(G)$ with $|S| = k$. Let $t, q, p \in \mathbf{N}$ with $q \geq 2$ and $p = \lceil \log_q(t+1) \rceil$. The following are equivalent:*

i. (Sum has target value.) We have $t = \sum_{e \in E(G[S])} w(e)$.

ii. (Expansions and carries satisfy linear constraints.) There is a sequence $c_0, c_1, \cdots \in \{0, \ldots, 2\binom{k}{\leq d}\}$ such that $c_0 = 0$ and the following linear equations hold for all $\ell \in \mathbf{N}$:

$$q c_{\ell+1} + t_\ell = c_\ell + \sum_{e \in E(G[S])} w_\ell(e)\,.$$

iii. (Expansions and carries satisfy a quadratic equation.) There is a sequence $c_0, c_1, \cdots \in \{0, \ldots, 2\binom{k}{\leq d}\}$ such that $c_0 = 0$ and the following quadratic equation holds:

$$\sum_{\ell \in \mathbf{N}} \Big(c_\ell - t_\ell - q c_{\ell+1} + \sum_{e \in E(G[S])} w_\ell(e) \Big)^2 = 0\,. \tag{2}$$

Proof. The equivalence between ii. and iii. follows from the fact that a sum of squares is zero if and only if all summands are zero. To see that i. implies ii., suppose $t = \sum_e w(e)$, so the q-expansions are the same as well: $t_\ell = \big(\sum_e w(e)\big)_\ell$ for all ℓ. We inductively set the carries so as to satisfy the linear equations; this choice is unique. It remains to argue that the c_ℓ are integers between 0 and $2\binom{k}{\leq d}$. The fact that they are non-negative integers is a standard property of q-expansions, so we only show the upper bound. We do so by induction: It clearly holds for c_0. In general, we have

$$c_{\ell+1} = -\frac{t_\ell}{q} + \frac{c_\ell}{q} + \sum_e \frac{w_\ell(e)}{q}\,.$$

The first summand $-t_\ell/q$ is at most 0, the second summand c_ℓ/q is at most $2\binom{k}{\leq d}/q \leq \binom{k}{\leq d}$ by induction and $q \geq 2$, and the third summand is at most $\binom{k}{\leq d}$, because $w_\ell(e) < q$ holds and the sum has at most $\binom{k}{\leq d}$ terms e.

To see that the second claim implies the first, we observe

$$\sum_{\ell \in \mathbf{N}} q^\ell \sum_{e \in E(G[S])} w_\ell(e) = \sum_\ell q^\ell (t_\ell + q c_{\ell+1} - c_\ell)$$
$$= t + \sum_{\ell > 0} q^\ell c_\ell - \sum_\ell q^\ell c_\ell = t - c_0 = t\,. \qquad \blacktriangleleft$$

The following algorithm uses (2) to reduce weights; in particular, we use the binomial theorem $(a+b)^2 = a^2 + 2ab + b^2$ in (2) (with $a = c_\ell - t_\ell - q c_{\ell+1}$) and then collect terms depending on which vertices of G the weight terms depend on – the terms not depending on edge weights are collected into the target integer. As discussed in the introduction, this approach was used before to reduce weights of cliques by Abboud et al. [4] in the more specific setting of *node weights* in graphs (rather than hypergraps).

Algorithm A (*Weight reduction for the weighted k-clique problem*) *Given a d-hypergraph G with edge weights $w : E(G) \to \mathbf{Z}$, a number k, a weight target $t \in \mathbf{Z}$, a parameter $p \in \mathbf{N}$, and access to an oracle for weighted k-clique in $2d$-hypergraphs, the following algorithm finds a k-clique of weight t in G:*

A1 *(Make k-partite and non-negative)* Apply Fact 3.1 to make the instance complete and k-partite and all weights non-negative.

A2 *(Set parameters)* Let $M = \max(\{t\} \cup \{w(e) : e \in E(G)\})$ and let $q \in \mathbf{N}$ be such that $p = \lceil \log_q M \rceil$.

A3 *(Guess carries)* Exhaustively guess $c_\ell \in \{1, \dots, 2\binom{k}{\leq d}\}$ for each $\ell \in \{1, \dots, p\}$; set $c_0 = 0$. For each such guess do the following:

a *(Compute new weights)* For every set $f \in \binom{V(G)}{\leq 2d}$, let

$$w'(f) = \sum_{\ell=0}^{p} \Big(2 \cdot [f \in E(G)] \cdot w_\ell(f) \cdot (c_\ell - t_\ell - q c_{\ell+1}) + \sum_{\substack{e_1, e_2 \in E(G) \\ e_1 \cup e_2 = f}} w_\ell(e_1) \cdot w_\ell(e_2) \Big)\,,$$
$$t' = -\sum_{\ell=0}^{p-1} (c_\ell - t_\ell - q c_{\ell+1})^2\,.$$

b *(Call oracle)* If the oracle detects a k-clique S in (G', w') of weight t', then halt and output *yes*; otherwise continue guessing carries.

A4 If no suitable carries were found, output *no*.

▶ **Lemma 3.4.** *Let $d, k \in \mathbf{N}$ with $1 \leq d \leq k$. Algorithm A (with input parameter $p \in \mathbf{N}$) is an oracle reduction from Weighted d-Hypergraph k-Clique to Weighted $2d$-Hypergraph k-Clique. The algorithm runs in time $O(p 4^d n^{2d} k^{dp})$ and makes at most k^{dp} oracle queries. Every query is a hypergraph on the same set of vertices. If M is the maximal weight among w and t, then the maximal weight M' of all queries satisfies $M' \leq O(k^{4d} M^{2/p} p)$.*

Proof. Let G be the d-hypergraph with $E(G) = \binom{V(G)}{\leq d}$, edge weight function $w : E(G) \to \mathbf{N}$, and target $t \in \mathbf{N}$ after applying Fact 3.1. Let G' be the $2d$-hypergraph with $E(G') = \binom{V(G)}{\leq 2d}$.

We first prove the correctness of the reduction. By Lemma 3.3, the instance (G, k, w, t) has a k-clique S of total weight t if and only if there exist $(c_\ell)_\ell$ satisfying (2). Now consider the weight of S in G'. We abbreviate the terms in the left side of (2) with $a_\ell = c_\ell - t_\ell - qc_{\ell+1}$ and $b_\ell = \sum_e w_\ell(e)$, and have

$$\sum_{\ell \in \mathbf{N}} (a_\ell + b_\ell)^2 = \sum_{\ell \in \mathbf{N}} \left(a_\ell^2 + 2a_\ell b_\ell + b_\ell^2\right) = \sum_{\ell \in \mathbf{N}} a_\ell^2 + \sum_{\ell \in \mathbf{N}} \left(2a_\ell b_\ell + b_\ell^2\right).$$

The squares of b_ℓ expand as follows:

$$b_\ell^2 = \left(\sum_e w_\ell(e)\right)^2 = \sum_{\substack{e_1, e_2 \in E(G) \\ e_1 \cup e_2 = f}} w_\ell(e_1) \cdot w_\ell(e_2).$$

Now we observe that t' and $w'(f)$ were defined exactly as to satisfy

$$\sum_{\ell \in \mathbf{N}} (a_\ell + b_\ell)^2 = -t' + \sum_{f \in E(G'[S])} w'(f).$$

By Lemma 3.3, the set S is a k-clique of weight t in (G, w) if and only if the right side of the latter equation is equal to 0, which in turn holds if and only if the weight of S with respect to w' is t'.

For the running time, note that the preprocessing takes $O(n^d)$ time. Exhaustive search for the carries takes $O(k^{dp})$ iterations, and each iteration takes time $O(n^{2d} p 4^d)$ because of line A3a in which we need to compute $w'(f)$ for every edge; overall the reduction takes time $O(n^{2d} k^{dp})$ and makes at most k^{dp} oracle queries.

For the weights, note $c_\ell \leq 2k^d$ and so $|t'| \leq O(k^{2d} q^2 p)$. To get the bound on $w'(f)$, observe that the term

$$\sum_{e_1 \cup e_1 = f} w_\ell(e_1) w_\ell(e_2)$$

is bounded by $4^d q^2$, and the term $w_\ell(f) \cdot (c_\ell - t_\ell - qc_{\ell+1})$ is bounded by $O(q^2 k^d)$ in absolute value. The preprocessing step relying on Fact 3.1 may have added an additional factor of k^{2d}; overall, all weights are bounded by $O(k^{4d} q^2 p)$. ◀

We apply Lemma 3.4 to reduce the maximum weights from $\mathrm{poly}(n)$ to $O(\log n)$, which is small enough to allow for exhaustive enumeration to reduce to the problem without weights.

▶ **Lemma 3.5.** *Let $d, k \in \mathbf{N}$ with $1 \leq d \leq k$ and $f(d, k) \in \mathbf{N}$. There is an $n^{2d+o(1)}$-time oracle reduction from Weighted d-Hypergraph k-Clique with weights in $\{-n^{f(k,d)}, \ldots, n^{f(k,d)}\}$ to unweighted k-partite $2d$-hypergraph k-Clique. If the input has n vertices, every oracle query has n vertices and the reduction uses at most $n^{o(1)}$ queries. Here the $o(1)$ terms are of the form $g(k, d)/\sqrt{\log n}$.*

Proof. Let G be a k-partite d-hypergraph with edge-weight function $w : E(G) \to \mathbf{Z}$ and target value $t \in \mathbf{Z}$. We apply Lemma 3.4. In particular, setting $p = \sqrt{\log n}$, we get $k^{dp} = n^{o(1)}$ queries and maximum weight $M' \leq O(k^{4d} M^{2/p} p) = n^{o(1)}$.

Each query is now a k-partite instance (G', w', k, t') with maximum weight M', where we treat k and d as constants. A solution S of G' satisfies $\sum_{e \in E(G'[S])} w'(e) = t'$. Since G' is k-partite, S intersects each part in exactly one vertex, and for each set $C \subseteq \{1, \ldots, k\}$ with

$1 \leq |C| \leq d$, there is a unique edge $e_C \in E(G'[S])$ that intersects exactly the color classes in C, and this edge contributes $w'(e_C)$ to the total weight of S. We want to simulate these weights by exhaustively guessing the contribution $w'(e_C)$ of each C. To do so, we only keep the edges of color type C that have the guessed weight.

More precisely, for each $C \subseteq \{1, \ldots, k\}$ with $1 \leq |C| \leq d$, we exhaustively guess a weight $a_C \in [-M', M']$. In total, this requires iterating through at most $(2M'+1)^{k^d} = n^{o(1)}$ candidate weight vectors $a = (a_C)_{C \subseteq \{1,\ldots,k\}}$. If the sum $\sum_C a_C$ is not equal to t', we reject the candidate vector and move to the next one. Otherwise, for each C and each edge e intersecting exactly the color classes prescribed by C, we keep e in the graph if and only if $w'(e) = a_C$. In this way, we obtain a k-partite $2d$-hypergraph G_a. For each candidate vector a of weight t', we query the (unweighted) k-clique oracle for $2d$-hypergraphs and output yes if and only if at least one query outputs yes.

The claim on the running time follows, since there are only $n^{o(1)}$ candidate vectors when k is regarded as a constant, and each oracle query G_a is prepared in time $n^{2d+o(1)}$, which is almost-linear in the description length of G_a. For the correctness, note that G has a solution S if and only if G' has a solution S. If G' has a solution S, then there is a setting of the a_C corresponding to the solution such that all edges in $G[S]$ survive in G_a, and the oracle finds a k-clique. On the other hand, if S is a k-clique in some G_a, then the used edges have the desired weight in G'. The correctness of the reduction follows. ◀

3.4 Reduction to Orthogonal Vectors

In this section, we reduce from k-Clique in d-hypergraphs via the k-OV problem to 2-OV. Recall that the k-OV problem is, given k sets $X_1, \ldots, X_k \subseteq \{0,1\}^D$ of Boolean vectors, to find $x_1 \in X_1, \ldots, x_k \in X_k$ such that $\sum_{j=1}^{D} \prod_{i=1}^{k} x_{ij} = 0$ holds, where the sum and product are the usual operations over the integers.

▶ **Lemma 3.6.** *Let $d, k \in \mathbf{N}$ with $1 \leq d \leq k$. There is a many-one reduction from (unweighted) k-partite d-hypergraph k-Clique to k-OV that runs in time $O(n^{d+1})$ polylog n; the number of produced vectors is n and the dimension of the vectors is n^d.*

Proof. Let G be a k-partite d-hypergraph with parts $V_1, \ldots, V_k$. Let $v_1, \ldots, v_k$ be vertices with $v_i \in V_i$ for all $i \in \{1, \ldots, k\}$. Then $\{v_1, \ldots, v_k\}$ forms a k-clique in G if and only if all non-edges $h \in \overline{E}(G)$ satisfy $h \not\subseteq \{v_1, \ldots, v_k\}$. Here $\overline{E}(G)$ denotes the set

$$\left\{ e \in \binom{V(G)}{\leq d} : \forall i . |e \cap V_i| \leq 1 \right\} \setminus E(G) .$$

We construct the instance $X_1, \ldots, X_k$ of k-OV as follows. For each $v \in V_i$, we create a vector $x_v \in X_i \subseteq \{0,1\}^{\overline{E}(G)}$ as follows: If $h \in \overline{E}(G)$ is disjoint from V_i, we set $x_{v,h} = 1$. If $h \cap V_i = \{v\}$, we set $x_{v,h} = 1$. Otherwise we have $h \cap V_i = \{u\} \neq \{v\}$ for some u, and we set $x_{v,h} = 0$. Clearly the sets $X_1, \ldots, X_k$ contain a total of n Boolean vectors, each with $|\overline{E}(G)| \leq n^d$ dimensions. Moreover, the sets are easily computed in $O(n^{d+1} \operatorname{polylog} n)$ time. It remains to prove the correctness of the reduction.

To this end, let $v_1, \ldots, v_k$ with $v_i \in V_i$ for all i be vertices that form a k-clique $\{v_1, \ldots, v_k\}$ in the k-partite d-hypergraph G. We claim that $\{x_{v_i}\}$ is a solution to the k-OV instance, that is, we claim $\sum_{h \in \overline{E}(G)} \prod_{i=1}^{k} x_{v_i,h} = 0$. To see this, let $h \in \overline{E}(G)$ be arbitrary. Since $\{v_1, \ldots, v_k\}$ is a k-clique in G and h is a non-edge of G, there exists a part V_i that satisfies $V_i \cap h \neq \emptyset$ and $v_i \notin h$. By definition, we have $x_{v_i,h} = 0$. Thus the entire sum is indeed zero.

For the reverse direction, let $x_{v_1}, \ldots, x_{v_k}$ with $x_{v_i} \in X_i$ for all i be vectors that form a solution to the k-OV instance. This means that for all $h \in \overline{E}(G)$, there exists some

$i \in \{1, \ldots, k\}$ such that $x_{v_i,h} = 0$ holds. By definition, this implies that $h \cap V_i = \{u\} \neq \{v_i\}$ for some u holds. Thus in particular, $h \not\subseteq \{v_1, \ldots, v_k\}$ and so the set $\{v_1, \ldots, v_k\}$ does not contain any non-edges of G and must be a clique. ◀

The last step of our reduction is reminiscent of the classic SETH-based lower bound for the 2-OV problem [54].

▶ **Lemma 3.7.** *Let $k \in \mathbf{N}$. There is an $O(n^{\lceil k/2 \rceil} D)$ time many-one reduction from k-OV to 2-OV that maps instances with n vectors in dimension D to instances with $O(n^{\lceil k/2 \rceil})$ vectors in dimension D.*

Proof. Let $X_1, \ldots, X_k \subseteq \{0,1\}^D$ be the input for k-OV with $n = \sum_{i=1}^{k} |X_i|$. The idea is to split the instance into two halves and list all candidate solutions in each half. For each candidate solution $S \subseteq X_1 \cup \cdots \cup X_{\lfloor k/2 \rfloor}$ with $|S \cap X_i| = 1$ for all $i \in \{1, \ldots, \lfloor k/2 \rfloor\}$, we create a vector $v^S \in X'_1 \subseteq \{0,1\}^D$ by setting $v^S_i = \prod_{u \in S} u_i$. Similarly, for each $S' \subseteq X_{\lfloor k/2 \rfloor + 1} \cup \cdots \cup X_k$ with $|S' \cap X_i| = 1$ for all $i \in \{\lfloor k/2 \rfloor + 1, \ldots, k\}$, we create a vector $v^S \in X'_2 \subseteq \{0,1\}^D$ by setting $v^S_i = \prod_{u \in S} u_i$. We obtain an instance $X'_1, X'_2 \subseteq \{0,1\}^D$ of 2-OV.

We claim that $X_1, \ldots, X_k$ is a yes-instance of k-OV if and only if X'_1, X'_2 is a yes-instance of 2-OV. Suppose that $v_1, \ldots, v_k$ are orthogonal, that is, $\prod_{i=1}^{k} (v_i)_j = 0$ holds for all $j \in \{1, \ldots, D\}$. We set $S = \{v_1, \ldots, v_{\lfloor k/2 \rfloor}\}$ and $S' = \{v_{\lfloor k/2 \rfloor + 1}, \ldots, v_k\}$. Clearly $v^S_j \cdot v^{S'}_j = 0$ holds for all j, so $v^S, v^{S'} \in V'$ are indeed orthogonal. Conversely, if $v^S_j \cdot v^{S'}_j = 0$ holds for all j, then the k vectors in $S \cup S'$ are orthogonal. ◀

3.5 Tying Things Together

We now formally prove Theorem 1.3.

▶ Theorem 1.3 (restated). If the moderate-dimension OV conjecture is false, then there exists an $\varepsilon > 0$ such that for every integer d there is a (large) integer $k = k(d, \varepsilon)$ satisfying the following statements:

- k-Clique can be solved on d-hypergraphs in time $O(n^{(1-\varepsilon)k})$.
- Exact-Weight-k-Clique and Min-Weight-k-Clique can be solved on d-hypergraphs with weights in $\{-M, \ldots, M\}$ in randomized time $O(n^{(1-\varepsilon)k}) \cdot \operatorname{polylog} M$.

Proof. Let (G, w, k, t) be an instance of Min-Weight-k-Clique on d-hypergraphs. We summarize the lemmas of this section as follows:

1. Lemma 3.2 randomly reduces in $\operatorname{polylog}(M)$ time from Exact-Weight-k-Clique with weights up to M to a constant (which depends on k and d) number of instances of Exact-Weight-k-Clique on G with weights up to $n^{O(k)}$.
2. Lemma 3.5 reduces this in $n^{2d+o(1)}$ time to $n^{o(1)}$ instances of k-Clique on $2d$-hypergraphs.
3. Lemma 3.6 reduces in $n^{2d+1+o(1)}$ time any instance of k-Clique on $2d$-hypergraphs to an instance of k-OV with n vectors in n^{2d} dimensions.
4. Lemma 3.7 reduces in time $n^{\lceil k/2 \rceil + 2d + o(1)}$ any such an instance of k-OV to an instance of 2-OV with $O(n^{\lceil k/2 \rceil})$ vectors in n^{2d} dimensions.

Composing the reductions gives a randomized $O(n^{\lceil k/2 \rceil + 2d + o(1)}) + \operatorname{polylog}(M)$ time oracle reduction from Exact-Weight-k-Clique on d-hypergraphs with n vertices and largest weight M to 2-OV on $n^{\lceil k/2 \rceil}$ vectors of dimension n^{2d} using $n^{o(1)}$ oracle calls.

To reduce from Min-Weight-k-Clique to Exact-Weight-k-Clique we use [43, Theorem 1], which allows us to perform a binary search for the minimum-weight-k-clique by making few queries to Exact-Weight-k-Clique. As the domain $E(G)$ of our weight function is of size $O(n^d)$ and the maximum weight is upper bounded by M, this reduction requires $O(n^d \log M)$ oracle calls. This yields a randomized oracle reduction from Min-Weight-k-Clique on d-hypergraphs with n vertices and largest weight M to 2-OV, which runs in time $O(n^{\lceil k/2 \rceil + 2d + o(1)}) \operatorname{polylog}(M)$ and makes $n^{d+o(1)} \log M$ oracle calls. To reduce from k-Clique on d-hypergraphs with n vertices to 2-OV, we only use steps 3 and 4, which takes time $O(n^{\lceil k/2 \rceil + 2d + o(1)})$.

All three reductions produce instances of 2-OV with N vectors in D dimensions, where $N = n^{\lceil k/2 \rceil}$ and $D = n^{2d} = N^\delta$ for some $\delta = \delta(k,d) > 0$. If k is large enough compared to d, then δ is arbitrarily close to zero. If the moderate-dimension OV conjecture is false, there exist $\delta, \varepsilon' > 0$ such that 2-OV with $D = N^\delta$ has an $O(N^{2-\varepsilon'})$ time algorithm. Combined with any of the three reductions, we obtain three algorithms that run in time at most

$$O\Big(n^{\lceil k/2 \rceil + 2d + o(1)} + n^{\lceil k/2 \rceil (2-\varepsilon') + d + o(1)}\Big) \operatorname{polylog}(M)\,.$$

When k is large enough compared to d, this is $O(n^{k(1-\varepsilon)}) \operatorname{polylog}(M)$ for some $\varepsilon = \varepsilon(\varepsilon') > 0$. This proves the claim. ◀

And we have the following corollary to Theorem 1.3.

▶ Corollary 1.4 (restated). If the moderate-dimension OV conjecture is false, then there exists an $\varepsilon > 0$ such that, for all integers d, there is an algorithm for Max-d-SAT that runs in time $O^*\big((2-\varepsilon)^n\big)$.

The corollary follows from Theorem 1.3 with a reduction from Max-d-SAT to k-Clique on d-hypergraphs that was already sketched in e.g. [54]. We formally state and prove this reduction next.

▶ **Lemma 3.8.** *Let $d, k \in \mathbf{N}$ with $1 \le d \le k$. There is an $O^*(2^{dn/k})$ time reduction from Max-d-SAT to Min-Weight-k-Clique on d-hypergraphs that maps d-CNF formulas with n variables and m clauses to d-hypergraphs with at most $k2^{n/k}$ vertices and integer edge weights between $-2m$ and $2m$.*

Proof. Given an instance of Max-d-SAT consisting of a d-CNF formula φ on variable set V of size n and m clauses, and an integer t indicating the required number of satisfied clauses, partition V into sets $V_1, \ldots, V_k$ where $|V_k| \le n/k$. The reduction computes from φ an instance H of Min-Weight-k-Clique.

We build a complete k-partite d-hypergraph H with vertices $\bigcup_{i=1}^k P_i$ where P_i contains a vertex p_x^i for every vector $x \in \{0,1\}^{V_i}$. Create an edge $f = \{x_1, \ldots, x_\ell\}$ for every set $\{i_1, \ldots, i_\ell\} \in \binom{[k]}{\le d}$ and tuple $(x_1, \ldots, x_\ell) \in P_{i_1} \times \ldots \times P_{i_\ell}$. Define the weight of f to be -1 times the number of clauses that

1. are contained in $\bigcup_{j=1}^\ell V_{i_j}$,
2. contain a variable in V_{i_j} for every $j = 1, \ldots, \ell$, and
3. are satisfied by the partial assignment obtained by setting the variables in V_{i_j} according to x_j.

The target instance is H, and the goal is to decide whether the minimum weight of any k-clique is at most $-t$. As the number of edges of H is at most $(k2^{n/k})^d$ and we compute their weights in polynomial time, the running time of this reduction is bounded by $O^*(2^{dn/k})$.

To see that this is a correct reduction, let $X \subseteq V(H)$ be a k-clique of H of weight at most $-t$. We see that X contains at most one vertex from every P_i, and as $|X| = k$ we have that X intersects in exactly one vertex with every P_i. By definition of the sets P_i, the set X thus corresponds to an assignment x of the variables of φ. We claim that the weight of X is -1 times the number of clauses satisfied by x and therefore φ has an assignment satisfying at least t clauses. To see the claim, let C be a clause of φ and $\{i : C \text{ contains variables from } V_i\} = \{i_1, \ldots, i_\ell\}$ be the set of variable groups intersecting C. Let $x_{i_1}, \ldots, x_{i_\ell}$ be the corresponding partial assignments that x induces to $V_{i_1}, \ldots, V_{i_\ell}$. We see that C contributes -1 to the weight of the hyperedge $(x_1, \ldots, x_\ell)$ and 0 to all other edges.

For the reverse direction, suppose x is an assignment satisfying at least t clauses of φ and let x_i be the projecting of x onto V_i. Then by the above claim the weight of $X := \{p^1_{x_1}, \ldots, p^k_{x_k}\}$ is $-t$. ◀

Proof of Corollary 1.4. For a sufficiently large constant k, we combine the reduction in Lemma 3.8 with an $O(n^{(1-\varepsilon)k})$ polylog M time algorithm for Min-Weight-k-Clique in d-hypergraphs. This yields an $O^*(2^{(1-\varepsilon)n})$ time algorithm for Max-d-SAT. Together with Theorem 1.3 this proves the claim. ◀

4 Reducing Sparse Satisfiability Problems to CNF-SAT

A dream theorem would be to reduce the sparse circuit satisfiability problem over the De Morgan basis to the CNF-SAT problem in such a way that a violation of SETH implies that faster algorithms for sparse circuit satisfiability exist. We demonstrate how to do this in Section 4.1 for sparse formulas as a warm-up, reproving a result of [28]. In Section 4.2, we prove Theorem 1.5, the extension of this result to sparse TC^0-circuits. We also prove that Theorem 1.6) for sparse TC^1-circuits and CNF-SAT in Section 4.3.

4.1 Sparse Formulas

Formulas are circuits that become a tree when the input gates are removed. We consider formulas over the De Morgan basis $\{\mathrm{NEG}, \mathrm{AND}, \mathrm{OR}\}$; in particular, we assume the corresponding trees to be binary, that is, the fan-in of every gate is at most two. We use the following simple decomposition lemma for binary trees:

▶ **Lemma 4.1** (Impagliazzo, Meka, and Zuckerman [33, Claim 4.4]). *Let T be a binary tree with m nodes and let $\ell \in \mathbf{N}$ with $\ell \leq m$. There exists a set $A \subseteq V(T)$ with $|A| \leq 6m/\ell$ such that every connected component $C \subseteq V(T)$ of $T - A$ has at most ℓ nodes and at most three vertices of A are adjacent to vertices of C. Moreover, such a set A can be computed in polynomial time.*

Using this lemma, the satisfiability of sparse Boolean formulas reduces to the satisfiability of k-CNF formulas with only a small overhead in the running time.

Algorithm B (*Transform sparse formula to k-CNF*) *Given a Boolean formula F and an positive integer $k \in \mathbf{N}$, this algorithm computes an equivalent k-CNF formula F'.*

B1 *(Compute decomposition)* Let $A \subseteq V(F)$ be the set guaranteed by Lemma 4.1 where $\ell = k/4$ and $T = F - I(F)$ is the tree obtained from F by removing its input gates.

B2 *(Create variables)* Let $x_1, \ldots, x_n$ be the input variables of F; for each $a \in A$, create a variable y_a. Also create a variable y_o where $o \in V(F)$ is the output gate of F.

B3 *(Compute k-CNFs for small subcircuits)* For each $v \in A \cup \{o\}$, do the following:

- **a** Let $F_{v,A}$ be the subcircuit of F induced by the set $R_C(A, v)$ (see (1) in the preliminaries) and interpret the gates $a \in A$ as input variables y_a.
- **b** Since $F_{v,A}$ depends on at most 2ℓ variables, we can compute a k-CNF formula F'_v with at most $2\ell + 1 \leq k$ variables that expresses the constraint "$y_v = F_{v,A}(x, y)$".

B4 *(Output)* Let $F' = y_o \wedge \bigwedge_{v \in A} F'_v$, and output F'.

We prove the correctness and properties of this algorithm.

▶ **Lemma 4.2.** *Let $c, \varepsilon > 0$. There exists a $k \in \mathbf{N}$ such that algorithm B is a polynomial-time many-one reduction for the satisfiability problem that maps formulas with n variables and at most cn gates to a k-CNF formula with at most $(1 + \varepsilon) \cdot n$ variables.*

Proof. We set $k = c/(24\varepsilon)$. Let F be the input formula with n variables $x_1, \ldots, x_n$ and $m \leq cn$ gates. We claim that F' has a satisfying assignment if and only if F does, and F' is a k-CNF formula with at most $(1 + \varepsilon)n$ variables.

Let T be the tree obtained when removing the input gates, and let $A \subseteq V(T)$ be the vertex set guaranteed by Lemma 4.1 for $\ell = k/4$. Since $m \leq cn$, we have $|A| \leq 24cn/k \leq \varepsilon n$, so indeed F' has at most $(1 + \varepsilon)n$ variables. By Lemma 4.1, $F_{v,A}$ contains at most ℓ non-input gates and, since the fan-in is at most two, $F_{v,A}$ contains most 2ℓ gates overall. So the constraint "$y_v = F_{v,A}(x, y)$" indeed depends on at most $2\ell + 1$ variables and can be expressed trivially by a $(2\ell + 1)$-CNF formula. It is clear that F' can be computed in polynomial time. Moreover, $F(x) = 1$ holds if and only if there is a setting for y such that $F'(x, y) = 1$ holds, so F and F' are equisatisfiable. We obtain the claimed reduction. ◀

Using this lemma, we prove that SETH is implied by an analogue of SETH for sparse formulas.

▶ **Theorem 4.3** (Dantsin and Wolpert [28])**.** *If SETH is false, then there is an $\varepsilon > 0$ such that, for all c, the satisfiability of Boolean formulas of size at most cn can be solved in time $O((2 - \varepsilon)^n)$.*

Proof. Suppose that SETH is false. Then there is some $\delta > 0$ such that k-CNF-SAT can be solved in time $O^*((2 - \delta)^n)$ for all $k \in \mathbf{N}$. Let $c > 0$. In order to solve Formula-SAT for cn-size formulas, we apply Lemma 4.2 to reduce to a k-CNF formula with $n' = (1 + \alpha)n$ variables. We can solve this instance using the assumed algorithm in time $O((2 - \delta)^{n'}) = O((2 - \delta)^{(1+\alpha)n}) = O((2 - \varepsilon)^n)$ for some suitable $\varepsilon, \alpha > 0$. ◀

4.2 Sparse TC0-circuits

The goal of this section is to prove the following:

▶ **Lemma 4.4.** *There is a polynomial-time many-one reduction from TC-SAT to CNF-SAT that, given $\varepsilon \in (0, 1)$ and a depth-d threshold circuit with at most cn wires, with $c \geq 1$, produces a k-CNF formula φ with at most $(1 + \varepsilon)n$ variables and width $k \leq \big(2000(c/\varepsilon)\log(2c/\varepsilon)\big)^d$.*

Our proof of Lemma 4.4 relies on a linear-size adder circuit as provided by the following lemma.

▶ **Lemma 4.5** (Adder circuit)**.** *Let $b, \ell \in \mathbf{N}$. There is a circuit $C_{add} : \{0,1\}^{b\ell} \to \{0,1\}^{b+\lceil \log \ell \rceil}$ over $\{\mathrm{NEG}, \mathrm{AND}, \mathrm{OR}\}$ with at most $40b\ell$ gates and maximum fan-in 2 such that C_{add} computes the binary representation of the sum of ℓ given b-bit integers.*

Proof. The proof of this lemma is standard. It is well known that there is a circuit C_{FA} (the full adder) that adds b'-bit numbers using $20b'$ gates and constant fan-in. Describing the computation of C_{add} using parentheses, the circuit C_{add} computes the sum $\sum_{i=1}^{\ell} b_i$ in a binary-tree-like way as

$$(((b_1 + b_2) + (b_3 + b_4)) + ((b_5 + b_6) + (b_7 + b_8))) + \dots$$

Using C_{FA} for every addition, the number of gates needed is at most

$$\sum_{i=1}^{\lceil \lg \ell \rceil} 20(b + i - 1)\ell/2^i \leq 20b\ell \sum_{i=1}^{\infty} i/2^i = 40b\ell \,. \qquad \blacktriangleleft$$

Our proof also needs a circuit computing the threshold function for binary inputs.

▶ **Lemma 4.6** (BINTH circuit)**.** *Let $r, \theta \in \mathbf{N}$. There is a circuit $\mathrm{BINTH}_\theta : \{0,1\}^r \to \{0,1\}$ over $\{\mathrm{NEG}, \mathrm{AND}, \mathrm{OR}\}$ with at most $2r$ gates and maximum fan-in 2 such that $\mathrm{BINTH}_\theta(x_0, \dots, x_{r-1})$ computes whether $\sum_{i=0}^{r-1} x_i 2^i$ is at least θ.*

Proof. The circuit BINTH_θ is constructed by setting $t = \lceil \lg \theta \rceil$ and converting the following expression to a circuit:
$\mathrm{BINTH}_\theta(x_0, \dots, x_{r-1}) =$

$$\Big(\bigvee_{i=t}^{r-1} x_i\Big) \vee \Big(x_{t-1} \wedge \mathrm{BINTH}_{\theta - 2^{t-1}}(x_0, \dots, x_{t-2})\Big) \,. \qquad \blacktriangleleft$$

We now have all tools needed to prove Lemma 4.4.

Proof of Lemma 4.4. Without loss of generality, threshold circuits only have input, NEG, and TH_θ gates, since TH_θ can directly simulate AND, OR, and MAJ gates. The algorithm to transform threshold circuits into k-CNF formulas is implemented in Algorithm C. Intuitively, it replaces threshold gates of large fan-in by a circuit of bounded fan-in in such a way that the circuit can be simulated by a k-CNF formula without introducing too many new variables.

Algorithm C (*Reduce sparse threshold circuit to k-CNF*) *Given a threshold circuit C of depth d and a positive integer $\beta \in \mathbf{N}$, this algorithm computes an equivalent k-CNF formula F.*

C1 *(Initialize gates to be replaced with variables)* Let $A = \{o\}$ where $o \in V(C)$ is the output gate of C.

C2 *(Replace large threshold gates by the circuit in Figure 2)* For each $v \in V(C)$ of with fanin $d_C^-(v) \geq \beta$:

- **a** Let $\theta \in \mathbf{N}$ such that v is a TH_θ-gate.
- **b** Partition the children $N_C^-(v)$ of v into blocks $B_1, \dots, B_\ell$ of size at most β with $\ell \leq \lceil d_C^-(v)/\beta \rceil$ and remove all wires leading into v.
- **c** *(Construct adder circuit for each block)* For each $i \in \{1, \dots, \ell\}$, create a circuit $C_{add}(B_i)$ that uses the gates of B_i as input gates and has $\log \beta + 1$ output gates b_i such that b_i represents the number of 1s in B_i in binary. Here, C_{add} is the circuit from Lemma 4.5.
- **d** *(Simulate threshold gate by circuit of fan-in two)* Add circuit $\mathrm{BINTH}_\theta(C_{add}(b_1, \dots, b_\ell))$ with inputs $b_1, \dots, b_\ell$ and output gate v, that is, the inputs $b_1, \dots, b_\ell$ are fed into C_{add} (from Lemma 4.5), whose outputs are fed into BINTH_θ (from Lemma 4.6). This concatenated circuit takes the binary representation of ℓ integers $(b_1, \dots, b_\ell) \in \{0, 1, \dots, \beta\}^\ell$

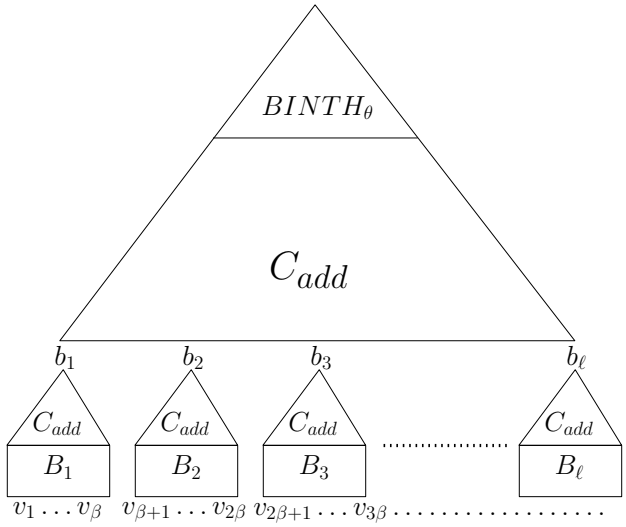

Figure 2 Overview of replacement of TH_θ gate.

on $b = \log \beta + 1$ bits each and it outputs true if and only if the sum of the given integers is at least θ. The circuit

$$\mathrm{BINTH}_\theta(C_{add}(b_1, \dots, b_\ell))$$

has at most $40b\ell + 2(b + \lceil \log \ell \rceil) \leq 44b\ell$ gates and fan-in at most 2. We add all of its gates, including the b_i's, to A.

C3 *(Compute k-CNFs)* For all $v \in A$, add a new variable y_v and do the following:

- **a** Let $C_{v,A}$ be the subcircuit of C induced by the set $R_C(A, v)$ (see (1) in the preliminaries) and interpret the gates $a \in A$ as input variables y_a.
- **b** We will show that $C_{v,A}$ depends on at most β^d variables, so we can compute a k-CNF formula F_v with at most $k = \beta^d$ variables that expresses the constraint "$y_v = C_{v,A}(x, y)$", where $x_1, \dots, x_n$ are the input variables of C.

C4 *(Output)* Let $F = y_o \wedge \bigwedge_{v \in A} F_v$, and output F.

Correctness of rewriting

To prove correctness, let us first observe that the transformation in C2 does not change the functionality of C since we just explicitly simulate threshold gates with large fan-in by constructing a small Boolean circuit over the De Morgan basis in a straightforward way. We do this transformation in a block-wise fashion in order to save the number of additional variables we add in step C3. So let C be the circuit after its transformation in C2 and consider step C3. Let x be a setting for the n input gates $I(C)$. We claim that $C(x) = 1$ holds if and only if there is a setting for the y-variables such that $F(x, y) = 1$. Indeed, if $C(x) = 1$, we set y such that $y_a = C_a(x)$ holds for all $a \in A$. This setting for y then satisfies the constraint "$y_v = C_{v,A}(x, y)$" and thus $F_v(x, y) = 1$. Moreover, $y_o = C(x) = 1$ holds as well, and so the formula F constructed in C4 is satisfied by (x, y). For the reverse direction, let (x, y) be such that $F(x, y) = 1$. We claim that $C(x) = 1$ holds as well. Indeed, by construction of F, we know that $y_o = 1$ and $C_{v,A}(x) = y_v$ holds for all $v \in A$. We can

see by induction on the depth of v (starting at the bottom) that $C_v(x) = C_{v,A}(x, y)$ holds. In the base case, the only input gates of $C_{v,A}$ are the original x-input gates from $I(C)$, and thus $y_v = C_{v,A}(x, y) = C_v(x)$ holds. In the inductive case, $C_{v,A}$ may depend on variables y_a. However, for each such variable, we know by the induction hypothesis that $y_a = C_{a,A}(x, y) = C_a(x)$ holds, and thus we have $C_v(x) = C_{v,A}(x, y)$ by the definition of C_v. In particular $C(x) = y_o = 1$ holds and so x satisfies C. This establishes the correctness of the reduction, except for proving that width $k = \beta^d$ is sufficient (in step C3b), which we will show next.

Bounding the width

In C3a, note that by the definition of $C_{v,A}$ and $R_C(A, v)$ (see Section 2), the value of gate v on input x is determined by the set of values of the gates $u \in (I(C) \cup A) \cap R_C(A, v)$. Therefore, in C3b we can ensure the value y_v equals the value of gate v on input x by adding clauses on y_v and the variables corresponding to the gates in $(I(C) \cup A) \cap R_C(A, v)$. We need to prove that the set $(I(C) \cup A) \cap R_C(A, v)$ has size less than $k = \beta^d$ so that this can be done in k-CNF. For most gates v added to A this is clear because there are only two gates feeding into v after the replacement step C2d and both of these gates are in A as well. The only exceptions are the $b_1, \dots, b_\ell$-gates. Note that any b_i-gate v is determined by the B_i-gates below it, so we can bound $|(I(C) \cup A) \cap R_C(A, v)| \leq \sum_{u \in B_i} |(I(C) \cup A) \cap R_C(A, u)|$. Any gate $u \in B_i$ already existed in the original circuit. If u has degree at least β in the original circuit, then we ran step C2 on u and thus u belongs to A. Otherwise, u has less than β children, which already existed in the original circuit, and on which we recurse. It follows that if gate u has depth d_u in the original circuit, then it can be reached from less than β^{d_u} nodes in A without going through any other node in A, i.e., $|(I(C) \cup A) \cap R_C(A, u)| < \beta^{d_u}$. Since u is a descendant of v, we have $d_u < d$. In total, we obtain $|(I(C) \cup A) \cap R_C(A, v)| < |B_i| \cdot \beta^{d-1} < \beta^d$. It follows that the constraints in step C3b indeed consider at most $k = \beta^d$ variables and can thus be expressed in k-CNF.

Bounding the number of variables

It remains to set β in such a way that $|A|$ is at most εn, which implies that F has at most $(1 + \varepsilon)n$ variables. Recall that the loop at C2 iterates over all gates v with fan-in $d_C^-(v) \geq \beta$. For any such gate v, we add at most $50b\ell$ gates to A (see step C2d), where $b = \log \beta + 1$ and $\ell \leq 1 + d_C^-(v)/\beta \leq 2d_C^-(v)/\beta$ since $d_C^-(v) \geq \beta$. Hence, overall the number of gates ever added to A is at most

$$\sum_{\substack{v \in V(G) \\ d_C^-(v) \geq \beta}} 100\, d_C^-(v) \cdot \frac{\log \beta + 1}{\beta} \leq 100\, cn \frac{\log \beta + 1}{\beta}\,.$$

Thus we can set $\beta = \beta(c, \epsilon) \in \mathbf{N}$ as a function of c and ϵ such that the size of A is at most ϵn. One can check that $\beta \leq 2000(c/\varepsilon) \log(2c/\varepsilon)$ suffices for $\varepsilon \leq 1 \leq c$, where all logs are base 2. Thus, we get the claimed upper bound on k. This concludes the proof of the lemma. ◀

Let us remark that in Lemma 4.4 we can also handle several other gates, such as MOD_m gates, by replacing the BINTH_θ circuit in the proof with a circuit that checks whether a given integer is a multiple of a given m. In fact, we can handle any symmetric gate $f(x_1, \dots, x_d) = g(\sum_{i=1}^d x_i)$ where $g(s)$ can be expressed as a $o(d)$-size DeMorgan circuit when given $s \in \{0, \dots, d\}$ in binary.

Using Lemma 4.4 for an ε with $(1 + \varepsilon)s_\infty < 1$ we obtain:

▶ Theorem 1.5 (restated). If SETH fails, then there is an $\varepsilon > 0$ such that, for all constants c and d, the satisfiability of depth-d threshold circuits with cn wires can be determined in time $O^*\big((2-\varepsilon)^n\big)$.

4.3 Improving the Dependence in Depth to Sub-exponential

In this section we improve the dependence of k on c, ε and d in Lemma 4.4. As this dependence is exponential in d, it is natural to employ existing techniques for depth reduction of circuits, such as the following result due to Valiant [52]. We use the following variant [37, Lemma 1.4] (see also [53, Section 4.2]).

▶ **Lemma 4.7.** *In any directed graph with m edges and depth 2^δ (where δ is integral), it is possible to remove in polynomial time a set R of rm/δ edges so that the depth of the resulting graph does not exceed $2^{\delta-r}$.*

We use Lemma 4.7 to improve the dependence of k in Lemma 4.4.

▶ **Lemma 4.8.** *There is an algorithm that, given $\varepsilon > 0$ and a depth-d threshold circuit with at most cn wires, produces at most $2^{\varepsilon n/2}$ k-CNF formulas $\varphi_1, \ldots, \varphi_z$ on $(1+\varepsilon/2)n$ variables such that the circuit C is satisfiable if and only if φ_i is satisfiable for some $i \le z$, and we have $k \le (4000(c/\varepsilon)\lg(4c/\varepsilon))^{(2d)^{1-\varepsilon/(2c)}}$.*

Proof. Let $C = (G = (V, E), \lambda)$ be the given circuit. Apply Lemma 4.7 to G with $\delta = \lceil \lg d \rceil$ and $r = \varepsilon\delta/(2c)$. We obtain a set R of size at most $\varepsilon n/2$ such that $(V, E \setminus R)$ has depth at most $2^{\delta(1-\varepsilon/(2c))} \le (2d)^{1-\varepsilon/(2c)}$. For every assignment $a \in \{0,1\}^R$ we create a circuit where we require that $C_x(v) = a_v$ for every $v \in R$, remove their outgoing wires, and update their incident gate accordingly (i.e. if $a_v = 1$ and the incident gate is a TH_θ it becomes a $\mathrm{TH}_{\theta-1}$ gate). The obtained circuit has depth at most $(2d)^{1-\varepsilon/(2c)}$ and applying Lemma 4.4 gives the claimed result. ◀

The improved dependence of k in Lemma 4.8 allows for the following consequence, yielding an exponential speedup for sparse threshold circuits of any depth $(\log n)^{1+o(1)}$.

▶ Theorem 1.6 (restated). If CNF-SAT can be solved in $O^*(2^{(1-\varepsilon)n})$ time for some $\varepsilon > 0$, then there is an $\varepsilon' > 0$ such that, for all $c > 0$, there is a $\delta > 0$ such that the satisfiability for threshold circuits of depth $(\log n)^{1+\delta}$ and at most cn wires can be determined in time $O(2^{(1-\varepsilon')n})$.

Proof. Apply Lemma 4.8 with $\varepsilon/2$ to obtain $2^{\varepsilon n/2}$ k-CNF formulas on $(1+\varepsilon/2)$ variables and with $k = \exp\big(O((\lg n)^{(1+\delta)(1-\varepsilon/(2c))})\big)$. For sufficiently small $\delta = \delta(\varepsilon, c) > 0$ we have $k = 2^{o(\log n)} = o(n/\log n)$ and thus the number of clauses m is at most $(2n)^k = n^{o(n/\log n)} = 2^{o(n)}$. Therefore the assumed algorithm for CNF-SAT determines the satisfiability of the produced CNF formula in time $2^{\varepsilon n/2} \cdot m^{O(1)} \cdot 2^{(1-\varepsilon)(1+\varepsilon/2)n}$, which is $O(2^{(1-\varepsilon')n})$ for any $\varepsilon' < \varepsilon^2/2$. ◀

Open Question.

It is known that Lemma 4.7 cannot be significantly improved (see [51]). However, this does not stop us from using the power of branching to get improvements. Specifically, when we try an assignments of the truth-value on edges in R in Lemma 4.4, all gates that are not connected to inputs are constant so these and their wires can already be computed and removed from the circuit. A natural question is whether this can be exploited more: Given a

DAG $G = (V, E)$ of depth 2^δ on m edges and a real number $0 < \alpha < 0$. For a set R of edges, denote $l(R)$ as the length of the longest path in $(V, E \setminus R)$ starting at a vertex v which is a source in G. Give an upper bound on $\min\{l(R) : |R| \leq \varepsilon m\}$ better than $2^{(1-\varepsilon)\delta}$ (which is implied by Lemma 4.7).

Acknowledgments

Jesper Nederlof is supported by NWO Veni grant 639.021.438. We thank Ivan Mikhailin for a helpful comment on an earlier version of this manuscript.

References

1 Amir Abboud, Arturs Backurs, Karl Bringmann, and Marvin Künnemann. Fine-grained complexity of analyzing compressed data: Quantifying improvements over decompress-and-solve. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 192–203. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.26.

2 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight hardness results for LCS and other sequence similarity measures. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 59–78. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.14.

3 Amir Abboud, Fabrizio Grandoni, and Virginia Vassilevska Williams. Subcubic equivalences between graph centrality problems, APSP and diameter. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 1681–1697. SIAM, 2015. doi:10.1137/1.9781611973730.112.

4 Amir Abboud, Kevin Lewi, and Ryan Williams. Losing weight by gaining edges. In Andreas S. Schulz and Dorothea Wagner, editors, *Algorithms - ESA 2014 - 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings*, volume 8737 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2014. doi:10.1007/978-3-662-44777-2_1.

5 Amir Abboud, Aviad Rubinstein, and R. Ryan Williams. Distributed PCP theorems for hardness of approximation in P. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 25–36. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.12.

6 Amir Abboud and Virginia Vassilevska Williams. Popular conjectures imply strong lower bounds for dynamic problems. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 434–443. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.53.

7 Amir Abboud, Virginia Vassilevska Williams, and Oren Weimann. Consequences of faster alignment of sequences. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2014. doi:10.1007/978-3-662-43948-7_4.

8 Amir Abboud, Virginia Vassilevska Williams, and Huacheng Yu. Matching triangles and basing hardness on an extremely popular conjecture. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 41–50. ACM, 2015. doi:10.1145/2746539.2746594.

9 Arturs Backurs, Nishanth Dikkala, and Christos Tzamos. Tight hardness results for maximum weight rectangles. In Ioannis Chatzigiannakis, Michael Mitzenmacher, Yuval Rabani, and Davide Sangiorgi, editors, *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, volume 55 of *LIPIcs*, pages 81:1–81:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2016. doi:10.4230/LIPIcs.ICALP.2016.81.

10 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In *Proc. of the 47th Annual ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 51–58, 2015.

11 Arturs Backurs and Piotr Indyk. Which regular expression patterns are hard to match? In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 457–466. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.56.

12 Arturs Backurs, Piotr Indyk, and Ludwig Schmidt. On the fine-grained complexity of empirical risk minimization: Kernel methods and neural networks. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 4311–4321, 2017.

13 Arturs Backurs and Christos Tzamos. Improving viterbi is hard: Better runtimes imply faster clique algorithms. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*, volume 70 of *Proceedings of Machine Learning Research*, pages 311–321. PMLR, 2017. URL: http://proceedings.mlr.press/v70/backurs17a.html.

14 Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Average-case fine-grained hardness. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 483–496. ACM, 2017. doi:10.1145/3055399.3055466.

15 Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh, editors. *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*. IOS Press, 2009.

16 Karl Bringmann. Why walking the dog takes time: Fréchet distance has no strongly subquadratic algorithms unless SETH fails. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 661–670. IEEE Computer Society, 2014. doi:10.1109/FOCS.2014.76.

17 Karl Bringmann, Pawel Gawrychowski, Shay Mozes, and Oren Weimann. Tree edit distance cannot be computed in strongly subcubic time (unless APSP can). In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 1190–1206. SIAM, 2018. doi:10.1137/1.9781611975031.77.

18 Karl Bringmann, Allan Grønlund, and Kasper Green Larsen. A dichotomy for regular expression membership testing. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 307–318. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.36.

19 Karl Bringmann and Marvin Künnemann. Quadratic conditional lower bounds for string problems and dynamic time warping. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 79–97. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.15.

20 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. A duality between clause width and clause density for SAT. In *21st Annual IEEE Conference on Computational Complexity (CCC 2006), 16-20 July 2006, Prague, Czech Republic*, pages 252–260. IEEE Computer Society, 2006. doi:10.1109/CCC.2006.6.

21 Timothy M. Chan and Ryan Williams. Deterministic apsp, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1246–1255. SIAM, 2016. doi:10.1137/1.9781611974331.ch87.

22 Ruiwen Chen and Rahul Santhanam. Improved algorithms for sparse MAX-SAT and MAX-k-CSP. In *Theory and Applications of Satisfiability Testing - SAT 2015 - 18th International Conference, Austin, TX, USA, September 24-27, 2015, Proceedings*, pages 33–45, 2015. doi:10.1007/978-3-319-24318-4_4.

23 Marek Cygan, Holger Dell, Daniel Lokshtanov, Dániel Marx, Jesper Nederlof, Yoshio Okamoto, Ramamohan Paturi, Saket Saurabh, and Magnus Wahlström. On problems as hard as CNF-SAT. *ACM Trans. Algorithms*, 12(3):41:1–41:24, 2016. doi:10.1145/2925416.

24 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

25 Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Fast Hamiltonicity checking via bases of perfect matchings. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 301–310. ACM, 2013. doi:10.1145/2488608.2488646.

26 Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In Rafail Ostrovsky, editor, *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS 2011, Palm Springs, CA, USA, October 22-25, 2011*, pages 150–159. IEEE Computer Society, 2011. doi:10.1109/FOCS.2011.23.

27 Søren Dahlgaard. On the hardness of partially dynamic graph problems and connections to diameter. 55:48:1–48:14, 2016. doi:10.4230/LIPIcs.ICALP.2016.48.

28 Evgeny Dantsin and Alexander Wolpert. Exponential complexity of satisfiability testing for linear-size boolean formulas. In Paul G. Spirakis and Maria J. Serna, editors, *Algorithms and Complexity, 8th International Conference, CIAC 2013, Barcelona, Spain, May 22-24, 2013. Proceedings*, volume 7878 of *Lecture Notes in Computer Science*, pages 110–121. Springer, 2013. doi:10.1007/978-3-642-38233-8_10.

29 Friedrich Eisenbrand and Fabrizio Grandoni. On the complexity of fixed parameter clique and dominating set. *Theoretical Computer Science*, 326(1):57–67, 2004.

30 Anka Gajentaan and Mark H. Overmars. On a class of $O(n^2)$ problems in computational geometry. *Comput. Geom.*, 5:165–185, 1995. doi:10.1016/0925-7721(95)00022-2.

31 François Le Gall. Faster algorithms for rectangular matrix multiplication. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 514–523. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.80.

32 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. In *Proc. of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2162–2181, 2017.

33 Russell Impagliazzo, William Matthews, and Ramamohan Paturi. A satisfiability algorithm for AC^0. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium*

on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012, pages 961–972, 2012. URL: http://portal.acm.org/citation.cfm?id=2095193&CFID=63838676&CFTOKEN=79617016.

34 Russell Impagliazzo and Ramamohan Paturi. On the complexity of k-SAT. *J. Comput. Syst. Sci.*, 62(2):367–375, 2001. doi:10.1006/jcss.2000.1727.

35 Russell Impagliazzo, Ramamohan Paturi, and Stefan Schneider. A satisfiability algorithm for sparse depth two threshold circuits. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 479–488, 2013. doi:10.1109/FOCS.2013.58.

36 Hamid Jahanjou, Eric Miles, and Emanuele Viola. Local reductions. In Magnús M. Halldórsson, Kazuo Iwama, Naoki Kobayashi, and Bettina Speckmann, editors, *Automata, Languages, and Programming - 42nd International Colloquium, ICALP 2015, Kyoto, Japan, July 6-10, 2015, Proceedings, Part I*, volume 9134 of *Lecture Notes in Computer Science*, pages 749–760. Springer, 2015. doi:10.1007/978-3-662-47672-7_61.

37 Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Springer Publishing Company, Incorporated, 2012.

38 Tsvi Kopelowitz, Seth Pettie, and Ely Porat. Higher lower bounds from the 3sum conjecture. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1272–1287. SIAM, 2016. doi:10.1137/1.9781611974331.ch89.

39 Robert Krauthgamer and Ohad Trabelsi. Conditional lower bounds for all-pairs max-flow. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 20:1–20:13. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.20.

40 Marvin Künnemann, Ramamohan Paturi, and Stefan Schneider. On the fine-grained complexity of one-dimensional dynamic programming. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, volume 80 of *LIPIcs*, pages 21:1–21:15. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.ICALP.2017.21.

41 Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Known algorithms on graphs on bounded treewidth are probably optimal. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 777–789. SIAM, 2011. doi:10.1137/1.9781611973082.61.

42 Daniel Moeller, Ramamohan Paturi, and Stefan Schneider. Subquadratic algorithms for succinct stable matching. In Alexander S. Kulikov and Gerhard J. Woeginger, editors, *Computer Science - Theory and Applications - 11th International Computer Science Symposium in Russia, CSR 2016, St. Petersburg, Russia, June 9-13, 2016, Proceedings*, volume 9691 of *Lecture Notes in Computer Science*, pages 294–308. Springer, 2016. doi:10.1007/978-3-319-34171-2_21.

43 Jesper Nederlof, Erik Jan van Leeuwen, and Ruben van der Zwaan. Reducing a target interval to a few exact queries. In Branislav Rovan, Vladimiro Sassone, and Peter Widmayer, editors, *Mathematical Foundations of Computer Science 2012 - 37th International Symposium, MFCS 2012, Bratislava, Slovakia, August 27-31, 2012. Proceedings*, volume 7464 of *Lecture Notes in Computer Science*, pages 718–727. Springer, 2012. doi:10.1007/978-3-642-32589-2_62.

44 Jaroslav Nešetřil and Svatopluk Poljak. On the complexity of the subgraph problem. *Commentationes Mathematicae Universitatis Carolinae*, 26(2):415–419, 1985.

45 Mihai Pătraşcu. Towards polynomial lower bounds for dynamic problems. In Leonard J. Schulman, editor, *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 603–610. ACM, 2010. doi:10.1145/1806689.1806772.

46 Mihai Pătraşcu and Ryan Williams. On the possibility of faster SAT algorithms. In Moses Charikar, editor, *Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2010, Austin, Texas, USA, January 17-19, 2010*, pages 1065–1075. SIAM, 2010. doi:10.1137/1.9781611973075.86.

47 Liam Roditty and Virginia Vassilevska Williams. Fast approximation algorithms for the diameter and radius of sparse graphs. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 515–524. ACM, 2013. doi:10.1145/2488608.2488673.

48 Liam Roditty and Uri Zwick. On dynamic shortest paths problems. In Susanne Albers and Tomasz Radzik, editors, *Algorithms - ESA 2004, 12th Annual European Symposium, Bergen, Norway, September 14-17, 2004, Proceedings*, volume 3221 of *Lecture Notes in Computer Science*, pages 580–591. Springer, 2004. doi:10.1007/978-3-540-30140-0_52.

49 Barna Saha. Language edit distance and maximum likelihood parsing of stochastic grammars: Faster algorithms and connection to fundamental graph problems. In Venkatesan Guruswami, editor, *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*, pages 118–135. IEEE Computer Society, 2015. doi:10.1109/FOCS.2015.17.

50 Rahul Santhanam and Srikanth Srinivasan. On the limits of sparsification. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 774–785. Springer, 2012. doi:10.1007/978-3-642-31594-7_65.

51 Georg Schnitger. A family of graphs with expensive depth reduction. *Theoretical Computer Science*, 18:89–93, 1982. doi:10.1016/0304-3975(82)90113-X.

52 Leslie G. Valiant. Graph-theoretic arguments in low-level complexity. In Jozef Gruska, editor, *Mathematical Foundations of Computer Science 1977, 6th Symposium, Tatranska Lomnica, Czechoslovakia, September 5-9, 1977, Proceedings*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977. doi:10.1007/3-540-08353-7_135.

53 Emanuele Viola. On the power of small-depth computation. *Foundations and Trends in Theoretical Computer Science*, 5(1):1–72, 2009. doi:10.1561/0400000033.

54 Ryan Williams. A new algorithm for optimal 2-constraint satisfaction and its implications. *Theoretical Computer Science*, 348(2-3):357–365, 2005. doi:10.1016/j.tcs.2005.09.023.

55 Ryan Williams. Improving exhaustive search implies superpolynomial lower bounds. *SIAM J. Comput.*, 42(3):1218–1244, 2013. doi:10.1137/10080703X.

56 Ryan Williams. Faster all-pairs shortest paths via circuit complexity. In David B. Shmoys, editor, *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 664–673. ACM, 2014. doi:10.1145/2591796.2591811.

57 Ryan Williams. Nonuniform ACC circuit lower bounds. *J. ACM*, 61(1):2:1–2:32, 2014. doi:10.1145/2559903.

58 Virginia Vassilevska Williams. Hardness of easy problems: Basing hardness on popular conjectures such as the strong exponential time hypothesis (invited talk). In Thore Husfeldt and Iyad A. Kanj, editors, *10th International Symposium on Parameterized and Exact Computation, IPEC 2015, September 16-18, 2015, Patras, Greece*, volume 43 of *LIPIcs*, pages 17–29. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2015. doi:10.4230/LIPIcs.IPEC.2015.17.

59 Virginia Vassilevska Williams and Ryan Williams. Subcubic equivalences between path, matrix and triangle problems. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 645–654. IEEE Computer Society, 2010. doi:10.1109/FOCS.2010.67.

60 Virginia Vassilevska Williams and Ryan Williams. Finding, minimizing, and counting weighted subgraphs. volume 42, pages 831–854, 2013. doi:10.1137/09076619X.

A Schematic Overview of our Results

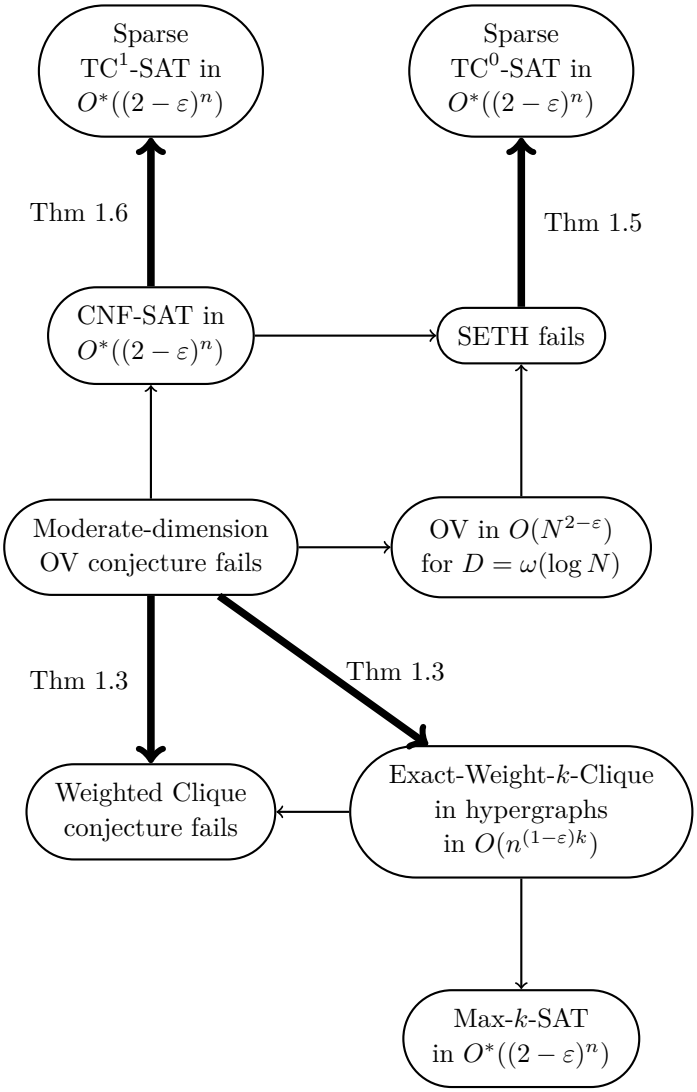

Figure 3 An overview of relevant implications. New implications presented in this paper are displayed with bold arcs and labeled with the theorem number.