

# New and Improved Concrete Lower Bounds for Orthogonal Vectors

Tameem Choudhury ✉ 

IIT Hyderabad, Hyderabad, India

Nutan Limaye ✉ 

IT University of Copenhagen, Copenhagen, Denmark

Karteek Sreenivasaiah ✉ 

University of Liverpool, Liverpool, UK

Srikanth Srinivasan ✉ 

University of Copenhagen, Copenhagen, Denmark

---

## Abstract

---

The Orthogonal Vectors Problem ( $\text{OV}_{n,d}$ ) takes as input two sets  $A, B$  each containing  $n$   $d$ -dimensional Boolean vectors, and outputs 1 if and only if there exists  $a \in A$  and  $b \in B$  such that  $a$  and  $b$  are orthogonal. The OV conjecture states that for every  $\varepsilon > 0$ , there exists a constant  $c \geq 1$  such that there is no algorithm deciding  $\text{OV}_{n,d}$  for  $d = c \log n$  with running time  $O(n^{2-\varepsilon})$ . The analogous kOV conjecture hypothesizes a lower bound of  $n^{k-\varepsilon}$  for the same problem with  $k$  sets. We prove these results and variants unconditionally in concrete computational models.

- We study a natural *monotone* version of the kOV-conjecture and shows that it holds for *monotone* circuits and *constant-depth* (not necessarily monotone) circuits when  $d = n^{\Omega(1)}$ .
- We show that the monotone version of the OV conjecture holds for *monotone circuits*. More formally, we show that for every  $\epsilon > 0$ , there exists  $c$  such that any monotone circuit family computing the negation of  $\text{OV}_{n,d}$  with  $d = c \log n$  must have size  $\Omega(n^{2-\epsilon})$ .
- We also prove stronger Boolean formula and branching program lower bounds for  $\text{OV}_{n,d}$ , strengthening a previous result of Kane and Williams (ITCS 2019). In particular, our Boolean formula lower bound of  $\Omega(n^2 d)$  is tight up to constant factors.

**2012 ACM Subject Classification** Theory of computation  $\rightarrow$  Circuit complexity; Theory of computation  $\rightarrow$  Problems, reductions and completeness

**Keywords and phrases** monotone circuit lower bounds, OV, approximation method

## 1 Introduction

**Motivation.** The Orthogonal Vectors ( $\text{OV}_{n,d}$ ) problem is the computational problem of deciding if among two given sets  $A$  and  $B$  of  $n$   $d$ -dimensional Boolean vectors, there exists a vector  $a \in A$  and a vector  $b \in B$  such that  $a$  and  $b$  are orthogonal. This problem has received a lot of attention over the past decade for being able to capture the hardness of several important computational problems that are seemingly unrelated. Some important examples of these include *edit distance* [9], *subset sum* [3], and *longest common subsequence* [1]. It has been established that if any of these problems has an algorithm that runs in truly sub-quadratic time, then so does OV. For a more thorough treatment of this line of work, we refer the reader to the survey by Vassilevska Williams [35]. This body of research has led to the following natural conjecture about the time complexity of deciding  $\text{OV}_{n,d}$ .

► **Conjecture 1** (OV conjecture). *For every  $\epsilon > 0$ , there exists  $c \geq 1$  such that there is no algorithm (deterministic or randomized) deciding  $\text{OV}_{n,d}$  for  $d = c \log n$  with running time  $O(n^{2-\epsilon})$ .*

In particular, when  $d \in \omega(\log n)$ , the conjecture states that there is no truly subquadratic algorithm deciding  $\text{OV}_{n,d}$ . It should be noted that when  $d < \log n$ , there are indeed subquadratic algorithms known [34]. Also, when  $d = c \log n$ , Abboud, Williams and Yu [4] show that  $\text{OV}_{n,d}$  can be decided by a randomized algorithm in time  $O(n^{2-1/O(\log c)})$ . This was later derandomized by Chan and Williams [17].

As positive evidence for the  $\text{OV}$  conjecture, Williams [33] showed that the Strong Exponential Time Hypothesis (SETH) [13, 23] implies the  $\text{OV}$  conjecture. Thus, proving the latter can also be seen as a stepping stone to SETH.

The fact that  $\text{OV}$  captures the hardness of a wide variety of problems makes it very interesting to ask if we can prove these conjectures unconditionally in restricted models of computation. The relevance of this line of research is twofold: the first, to rule out proposed algorithmic paradigms for these problems and the second, to find structural characteristics of the problem that will hopefully lead to an unrestricted lower bound in the future.

A recent line of research has been studying precisely this question. Kane and Williams [26] showed strong lower bounds for the size of Boolean Formulas and Branching Programs computing  $\text{OV}$  thus proving the  $\text{OV}$ -conjecture for these two models. Choudhury and Sreenivasiah [19] show that this conjecture is true when computation is restricted to depth-3  $\text{OR} \circ \text{AND} \circ \text{OR}$  circuits with constant bottom fan-in.

This latter work even proves such a result for the more general  $\text{kOV}$ -conjecture. In the  $\text{kOV}_{n,d}$  problem, the input is  $k$  lists  $A_1, \dots, A_k \subseteq \{0, 1\}^d$  of size  $n$  each and the question is to decide if there exist vectors  $a_1 \in A_1, \dots, a_k \in A_k$  such that  $a_1 \cap a_2 \cap \dots \cap a_k = \emptyset$  (here we identify each Boolean vector  $a_i \in \{0, 1\}^d$  with a subset of  $[d]$  in the natural way). The  $\text{kOV}$  conjecture, analogous to Conjecture 1 is as follows.

► **Conjecture 2** ( $\text{kOV}$  conjecture). *For every  $\epsilon > 0$  and  $k \geq 2$ , there exists  $c \geq 1$  such that there is no algorithm (deterministic or randomized) deciding  $\text{kOV}_{n,d}$  for  $d = c \log n$  with running time  $O(n^{k-\epsilon})$ .*

The complexity of  $\text{kOV}$  also has intimate connections with several important functions including graph diameter [10] and longest common subsequence [1]. (See [35] for more.)

**Results.** We now describe the results of this paper, which yield new and stronger statements along these lines.

**Monotone and constant-depth Boolean circuits.** A major focus of this paper is on *monotone* computation. Observe that the negation of  $\text{OV}$  (and more generally  $\text{kOV}$ ) is a monotone function.<sup>1</sup> We define the function  $\text{klnt}_{n,d} : (\{0, 1\}^d)^n \times (\{0, 1\}^d)^n \rightarrow \{0, 1\}$  as the negation of  $\text{kOV}_{n,d}$ .

► **Definition 3** ( $\text{klnt}_{n,d}$ ). *Fix any  $k \geq 2$ . For tuples  $A_1, A_2, \dots, A_k \subseteq \{0, 1\}^d$  with  $|A_1| = \dots = |A_k| = n$ ,*

$$\text{klnt}_{n,d}(A_1, \dots, A_k) = 1 \iff \forall a_1 \in A_1, \forall a_k \in A_k, \text{ we have } a_1 \cap a_2 \cap \dots \cap a_k \neq \emptyset.$$

(where we interpret each Boolean vector  $a_i$  as the characteristic vector of a subset of  $[d]$ .) In the particular case that  $k = 2$ , we use  $\text{Int}_{n,d}$  to denote  $\text{klnt}_{n,d}$ .

---

<sup>1</sup> A monotone function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is one that satisfies the property  $x \leq y \Rightarrow f(x) \leq f(y)$ .

In light of Conjectures 1 and 2, it is natural to study the complexity of  $\text{Int}_{n,d}$  and  $\text{klnt}_{n,d}$  for *monotone* circuits<sup>2</sup> and prove that any monotone Boolean circuit for this problem must have size at least  $n^{2-\epsilon}$  for  $d = c \log n$  (where  $c = c(\epsilon)$  is large enough). If Conjecture 1 is true for general Boolean circuits, then this monotone lower bound must also hold. On the other hand, we note that the power of monotone circuits is likely to be incomparable with non-monotone models such as depth-3 circuits, Boolean formulas and branching programs studied in earlier work. In particular, there are P-complete problems that have small monotone circuits and it is unlikely that these can be simulated by formulas or branching programs in polynomial size (this can be proved unconditionally for depth-3 circuits). This makes the monotone question interesting.

Another computational model we study in this paper is the class of *constant-depth circuits*.<sup>3</sup> This circuit class can implement many non-trivial algorithms such as the colour-coding algorithms for subgraph isomorphism [6] and is incomparable with models studied in previous work (and provably stronger than the depth-3 model studied in [18]).

Our first result is a proof of a weak form of Conjecture 2 (and thus also a weak form of Conjecture 1) in the above two settings. The reason this is a ‘weak form’ is that this proof works when the dimension  $d$  of the input vectors is  $n^{\Omega(1)}$ .

► **Theorem 4.** *Fix any constants  $k, h \geq 2$  and any constant  $\epsilon > 0$ . Any monotone circuit or any depth- $h$  (not necessarily monotone) circuit computing  $\text{klnt}_{n,d}$  for  $d = n^\epsilon$  must have size at least  $n^{k-\epsilon}$ .<sup>4</sup>*

The constant-depth circuit lower bound is a resolution (in a weaker form) of a question due to Paturi stated in the work of Kane and Williams [26].

While the above theorem only proves lower bounds for relatively large values of  $d$ , it is worth noting that the OV problem is already interesting for fine-grained complexity in this setting [22, 2].

Our next result is a complete resolution of the monotone Boolean circuit version of Conjecture 1.

► **Theorem 5.** *For every  $0 < \epsilon < 1$ , there exists a constant  $c$  such that for all  $d \geq c \log n$  any monotone circuit computing  $\text{Int}_{n,d}$  requires at least  $\Omega(n^{2-\epsilon})$  many gates.*

**Boolean Formulas and Branching Programs.** In our third result, we revisit the Boolean formula and Branching Program lower bounds of Kane and Williams [26]. We improve the parameters in both results. In particular, this leads to a *tight* lower bound (up to constant factors) for Boolean formulas over the full binary basis. More precisely, we show the following.

► **Theorem 6** (Boolean formula and Branching Program lower bound for  $\text{OV}_{n,d}$  (Informal)). *Assume  $d > c \log n$  for  $c > 1$ . Any binary Boolean formula computing  $\text{OV}_{n,d}$  has size at least  $\Omega(n^2 d)$ . Any Boolean branching program for  $\text{OV}_{n,d}$  has size at least  $\Omega(n^2 d / \log(nd))$ .*

We note that this bound is tight up to constant factors, as  $\text{OV}_{n,d}$  has a simple Boolean formula of size  $O(n^2 d)$ . Kane and Williams [26] prove a weaker lower bound of  $\Omega(n^2 / \log d)$

<sup>2</sup> That is, Boolean circuits made up of only AND and OR gates. Such circuits can compute all monotone functions.

<sup>3</sup> These are Boolean circuits made up of NOT gates and AND, OR gates of unbounded fan-in. The depth refers to the amount of nesting in the circuit, or alternatively the length of the longest input to output gate path.

<sup>4</sup> Since the constant-depth circuit model is not monotone, we could also have stated the lower bound for this model directly for  $\text{kOV}_{n,d}$ .

in this regime of parameters. For Branching programs Kane and Williams [26] show a lower bound of  $\Omega(n^2/(\log d \log nd))$ . Our result improves this bound by a factor of  $d \log d$ .

**Circuit constructions.** Kane and Williams [26] observed that the standard brute-force algorithms for kOV can be carried out by constant-depth circuits. We note that for  $\text{kInt}_{n,d}$ , these circuits can also be made monotone. We give two different brute-force constructions, one for small  $d$ , and another when  $d$  is large. In particular, for  $d \in o(\log n)$ , we show that a monotone formula of size  $O(2^d nd)$  suffices to compute  $\text{Int}_{n,d}$ . We note that for such small  $d$ , Williams [34] shows a  $\tilde{O}(1.35^d n)$  time randomized algorithm for OV. It is not clear if such a bound can be achieved using monotone circuits.

## 1.1 Techniques

**Lower bounds via reductions.** The proof of Theorem 4 is based on a simple split-and-list monotone reduction (inspired from [2]) from graph and hypergraph versions of the clique<sup>5</sup> problem to  $\text{kInt}_{n,d}$ . The basic observation is that a set  $S$  of vertices forms a clique of size  $\ell$  in a graph (or hypergraph) if and only if it can be partitioned into  $k$  subsets  $S_1, \dots, S_k$  of size  $\ell/k$  each such that each of them is a clique and each edge contained in the union  $S_1 \cup \dots \cup S_k$  is present in the graph (or hypergraph). This can be turned into a reduction from clique on  $m$ -vertex (hyper-)graphs by creating  $k$  lists of size  $\binom{m}{\ell/k}$  where each element of the  $i$ th list corresponds to a set  $S_i$  of  $\ell/k$  vertices and the corresponding vectors (which have dimension  $m^{O(1)}$  each) are orthogonal if and only if  $S_1 \cup \dots \cup S_k$  form a clique.

To obtain the required lower bound for  $\text{kInt}_{n,d}$  from here, we need a strong lower bound ( $m^{\ell \cdot (1-o(1))}$ ) for the clique problem on graphs or hypergraphs. In the monotone setting, such bounds have been known for a long time due to work of Razborov [31] (we use a statement from a paper of Alon and Boppana [5]). Similar results are known for hypergraph versions of the clique problem via a result of Amano [7]. Along with the reduction, these results imply the desired lower bound. The reduction and its consequences appear in Section 3.

Abboud, Bringmann, Dell and Nederlof [2] showed how to use the same high-level idea to reduce the problem of deciding  $r$ -partite clique on  $r$ -partite  $r$ -uniform hypergraphs to  $\text{kInt}_{n,d}$ .<sup>6</sup> Unfortunately, the requisite circuit lower bounds are not known in the  $r$ -partite setting and so we need to modify their reduction. The same work also gives reductions from the general  $r$ -uniform hypergraph clique problem to the  $r$ -partite case but it is unclear if such reductions can be implemented by monotone circuits.

**OV conjecture for monotone circuits.** The above reduction gives us the required lower bound for monotone circuits computing  $\text{Int}_{n,d}$  when the dimension  $d$  is larger than the range prescribed in Conjecture 1. The reasons for this have to do with existing monotone lower bounds for Clique not being tight in a fine-grained sense. Unfortunately, even the most optimistic monotone circuit lower bound for Clique (which is not known) would not yield a near quadratic lower bound for all  $d = \omega(\log n)$ . (This is described in more detail in Remark 16.) Thus, more ideas are required to prove the OV conjecture in its strong form for monotone circuits.

<sup>5</sup> The Clique problem asks if a given input graph or hypergraph (presented as an adjacency matrix or tensor) contains a clique of a given size.

<sup>6</sup> They state their results in terms of kOV $_{n,d}$  but we phrase it in terms of  $\text{kInt}_{n,d}$  as we prefer to stay with monotone problems.

To do this, we turn to known techniques for proving strong monotone circuit lower bounds, which have been known since the 1980s. Razborov [31] developed a method to prove superpolynomial lower bounds on the size of monotone circuits computing Clique. This technique is now called the ‘approximation method’. Since Razborov’s work, the approximation method has been the main approach to proving lower bounds against monotone circuits and has been refined and strengthened in several works such as [5, 8]. A crucial ingredient in the original approximation method is the Sunflower Lemma by Erdős and Rado [21]. Recently, Rossman [32] defined a more relaxed version of Sunflowers, now commonly known as “robust Sunflowers” and showed average case lower bounds for computing Clique. This has also been used to strengthen worst case monotone size lower bounds [16], [14].

To achieve better results than the one we get from the above reduction (Section 4), we apply a modified version of Razborov’s approximation method directly to the  $\text{Int}_{n,d}$  problem. For readers not familiar with the standard approximation method, we describe it briefly here. The idea is to start with a monotone circuit  $C$  of size  $s$  computing the target function  $f$ , and show that  $C$  can be “approximated” closely using a small depth-two circuit (i.e. a DNF or a CNF)  $\tilde{F}$ . Here by ‘approximation’, we mean the behaviour of  $\tilde{F}$  with respect to  $C$  on two distributions  $\mathbf{D}_0$  and  $\mathbf{D}_1$  on 0-inputs and 1-inputs of  $f$  respectively. Then we show that on the one hand, any small depth-two circuit must have large error, say  $\delta$ , with respect to  $f$ . On the other hand, the DNF  $\tilde{F}$  that we constructed has error at most  $s\epsilon$  with respect to  $C$ . Then we can conclude  $s \geq \delta/\epsilon$ .

In the typical setting where the approximation method is used, the notion of ‘small’ for the depth-two circuits has to do with the number of terms/clauses, and their maximum width. Ensuring that the approximator  $\tilde{F}$  is small usually involves a closure operation that uses the Sunflower Lemma or something similar [31, 5, 32]. In our setting, all these are too demanding to be used. At a high level, this is because the marginal probabilities of our chosen distribution  $\mathbf{D}_0$  are too low for such techniques to be applicable. See Remark 18.

The main difference between the standard approximation method and ours is in the structure of the approximator. Instead of a DNF/CNF, our approximator is a conjunction of *arbitrary* monotone Boolean functions that are each “local” to any one vector among the  $2n$  vectors. i.e., each function takes inputs from at most one vector. We define this in Definition 19. This allows us to avoid expensive combinatorial closures such as the Sunflower lemma that are inapplicable. A crucial property that we maintain for our approximator is that each function (in the conjunction) errs on  $\mathbf{D}_1$  with a non-trivial probability. Maintaining this property requires a technical lemma (Lemma 22) that is similar in spirit to the Kruskal-Katona Theorem ([28, 27]) but in a much simpler setting where we work with a product distribution.

**Formula and Branching Program lower bounds.** The source of improvement in the formula and branching program lower bounds is the same, so we describe the formula case. Consider the  $\text{OV}_{n,d}$  problem with input lists  $A, B$ . The basic idea of Kane and Williams, which fits into a classical framework for such lower bounds due to Nechiporuk [29], is to show that for any formula  $F$  and each input vector  $a \in A$ , the bits of  $a$  label many leaves of  $F$ . This can be done by setting all the bits of the other elements of  $A$  to 1 and<sup>7</sup> then setting the other elements of  $B$  from a fixed subset  $X \subseteq \{0, 1\}^d$  of size  $n$ . For each subset  $X' \subseteq X$ , we get a restricted function on the bits of  $a$  by using exactly the elements of  $X'$  as assignments

<sup>7</sup> This step is actually not required in the Kane-Williams argument as they consider the version of  $\text{OV}$  where  $A = B$ . The arguments are similar in both cases.

for the elements of  $B$ . This gives  $2^n$  different ‘sub-functions’ on the bits of  $a$  leading (by a simple counting argument) to a lower bound of roughly  $n$  on the number of leaves labelled by the bits of  $a$ .

To improve this argument, our main observation is that we do not need to fix the subset  $X$  beforehand, and doing so unnecessarily reduces the number of sub-functions. Instead, by counting the number of subsets  $X$ , we obtain a bound of (roughly)  $2^{nd}$  instead. This leads to a stronger (and in fact optimal) formula lower bound.

These results are proved in Section 5.

## 1.2 Other related work

**kOV <sub>$n,d$</sub>  when  $d$  is large.** As mentioned above, the kOV <sub>$n,d$</sub>  problem is already interesting in the setting when  $d = n^{\Omega(1)}$ . This problem was introduced in work of Gao, Impagliazzo, Kolokolova and Williams [22] who studied its connections to the fine-grained complexity of a large class of problems defined by first-order logic formulas. This was also further studied in the aforementioned work of Abboud et al. [2] who made connections to weighted versions of hypergraph clique problems and optimization variants of SAT.

**Jukna’s criterion.** Another well-known route for proving monotone circuit lower bounds is via the monotone switching lemma formulated by Jukna [24] and Berg and Ulfberg [11]. Unfortunately, the criterion used for lower bounds in these works does not seem to be applicable to our problem. See Section 7 for a justification.

## 2 Preliminaries

For formal definitions of Boolean circuits, formulas, and branching programs, we refer the reader to a standard text such as Jukna [25].

For any  $x, y \in \{0, 1\}^d$ , we write  $x \leq y$  if  $\forall i, x_i \leq y_i$ .

► **Definition 7 (Monotone function).** *We say that a Boolean function  $f$  is monotone if  $\forall x, y \in \{0, 1\}^d$  such that  $x \leq y$ , we have  $f(x) \leq f(y)$ .*

We often interpret a  $d$ -dimensional vector  $u \in \{0, 1\}^d$  as the characteristic vector of a subset of  $[d]$ . For a vector  $u_i$ , we denote the  $j$ ’th bit with  $u_{ij}$ .

Recall that the dual of a Boolean function  $f : \{0, 1\}^n \rightarrow \{0, 1\}$  is a Boolean function  $f^* : \{0, 1\}^n \rightarrow \{0, 1\}$  defined by

$$f^*(x_1, \dots, x_n) := \neg f(\neg x_1, \dots, \neg x_n).$$

The following is standard.

► **Observation 8.** *The dual  $f^*$  of a monotone Boolean function  $f$  is also monotone. Furthermore, for any  $s$ ,  $f$  has a monotone circuit of size  $s$  if and only if  $f^*$  has a monotone circuit of size  $s$ .*

In particular, we will sometimes consider the dual of  $\text{Int}_{n,d}$ . The following is a consequence of the definition of  $\text{Int}_{n,d}$ .

► **Observation 9.** *For tuples  $A, B \subseteq \{0, 1\}^d$  with  $|A| = |B| = n$ ,*

$$\text{Int}_{n,d}^*(A, B) = 1 \iff \exists a \in A, \exists b \in B \text{ such that } a \cup b = [d].$$

► **Definition 10** ( $\ell$ -clique). *The problem  $\ell$ -clique is that of deciding if a given graph contains a clique (complete graph) on a subset of at least  $\ell$  vertices.*

*When the parameter  $\ell$  is not important, we will use **Clique** to refer to the  $\ell$ -clique problem.*

We will use  $\mu_p$  to denote the distribution where each bit is set independently at random to 1 with probability  $p$ .

### 3 Lower bounds via reductions from Hyperclique

To state the main result of this section, we start with a definition.

For parameters  $m, r, \ell$ , let  $\text{HypClique}_{r,\ell}^m : \{0,1\}^{\binom{m}{r}} \rightarrow \{0,1\}$  be a Boolean function that takes as input the characteristic vector of the edge set of an  $r$ -regular hypergraph on  $m$  vertices and outputs a 1 if and only if it has a hyperclique of size  $\ell$ .

► **Theorem 11.** *The following holds for any small enough constant  $\varepsilon > 0$  and positive integer constants  $r, k, h$ . Let  $m$  be a parameter and let  $\ell := \ell(m)$  be a non-decreasing function such that  $\ell = o(m)$ ,  $\ell$  is divisible by  $k$ . Suppose any monotone circuit (resp. depth- $h$  (not necessarily monotone) circuit) computing  $\text{HypClique}_{r,\ell}^m$  has size at least  $\binom{m}{\ell}^{1-\varepsilon}$ . Then for  $n = \binom{m}{\ell/k}$ , there exists  $d = O(m^r)$  such that any monotone circuit (resp. depth- $h$  circuit) for  $\text{kInt}_{n,d}$  on length- $n$  lists of vectors of dimension  $d$  has size at least  $n^{k \cdot (1-2\varepsilon)}$ .*

Before we prove the theorem above, we use it to derive the monotone lower bound and constant-depth lower bound for computing  $\text{kInt}_{n,d}$ .

#### Monotone circuit lower bound.

For the monotone lower bound, we will use the following lower bound stated by Alon and Boppana [5] following the work of Razborov [31].

► **Theorem 12** ([5, 31]). *Any monotone circuit solving  $\ell$ -Clique on graphs with  $m$  vertices has size at least  $\frac{m^\ell}{2^{O(\ell^2) \cdot (\log m)^\ell}}$ . In particular, for any large enough  $\ell = \ell(m) = o(\log m)$ , the lower bound is  $m^{\ell \cdot (1-o(1))}$ .*

► **Corollary 13.** *Fix any constant  $\epsilon > 0$ . Let  $n$  be a growing parameter and  $\epsilon > 0$  any constant. There is a function  $d(n) = \exp((\log n)^{1/2+o(1)}) = n^{o(1)}$  such that for infinitely many  $n$ , any monotone circuit computing  $\text{kInt}_{n,d}$  must have size at least  $n^{k \cdot (1-\epsilon)}$ .*

**Proof of Corollary 13.** The proof is an immediate consequence of Theorem 11 along with Theorem 12 above. More precisely, we fix  $\ell$  to be a growing parameter of  $m$  such that  $\ell = \log m / \log \log m$  and  $\ell$  is divisible by  $k$ . We will apply Theorem 11 with  $\delta = \epsilon/2$ .

Note that Theorem 12 yields a monotone circuit lower bound for clique of size  $m^{\ell \cdot (1-o(1))} \geq \binom{m}{\ell}^{1-\delta}$ . We set  $n = \binom{m}{\ell/k} = \exp((\log m)^{2-o(1)})$  and  $d \leq m^2 = \exp(O(\log m)) = \exp((\log n)^{1/2+o(1)})$ . For this setting of parameters, Theorem 11 implies a lower bound of  $n^{k(1-2\delta)} = n^{k(1-\epsilon)}$ . ◀

#### Constant-depth lower bound.

For the constant-depth lower bound, we use the following theorem by Amano [7]. We will need the following definition to state the theorem.

► **Theorem 14.** *For every constant  $\ell > r > 2$  every depth- $h$  circuit computing  $\text{HypClique}_{r,\ell}^m$  must have size at least  $\Omega(m^{\ell \cdot (1 - \frac{\log r + 2}{r-1})})$ . In particular, for constant  $r$ , the lower bound is  $m^{\ell \cdot (1-\delta)}$  for a constant  $\delta$ .*

Now, using Theorem 11 and Theorem 14, we get the following corollary for depth- $h$  circuits which proves Theorem 5.

► **Corollary 15.** *Fix any constant  $\epsilon > 0$ . Let  $n$  be a growing parameter and  $\epsilon > 0$  any constant. There is a function  $d(n) = \exp((\log n)^{1/2+o(1)}) = n^{o(1)}$  such that for infinitely many  $n$ , any depth- $h$  circuit computing  $\text{klnt}_{n,d}$  must have size at least  $n^{k \cdot (1-\epsilon)}$ .*

**Proof of Corollary 15.** For any fixed  $k$ , and for  $r$  at least  $\frac{1}{\delta} \cdot \log(1/\delta)$ , we choose  $\ell$  such that  $\ell$  is divisible by  $k$  and  $\ell$  is at least  $kr/2\delta$ . Note that, for this setting and for  $n = \binom{m}{\ell/k}$  we have  $d = n^{o(1)}$ . Now, we can invoke Theorem 11. Note that, we have a lower bound  $m^{\ell(1-\delta)}$  from [7], which is stronger than  $\binom{m}{\ell}^{(1-\delta)}$ , which we need to apply Theorem 11. We get a lower bound of  $n^{k(1-2\delta)}$ . This completes the proof. ◀

**Proof of Theorem 11.** The proof is the same for both monotone circuits and depth- $h$  circuits.

Let  $G = (V, E)$  be an  $r$ -uniform hypergraph on  $m$  vertices. Let  $x$  be the bit-vector that represents this hypergraph when given as input to a circuit. That is, the entries of  $x$  are labelled by sets  $e \in \binom{V}{r}$  and  $x_e = 1$  if and only if the edge  $e$  is present in the hypergraph  $G$ . Informally,  $x$  is the *adjacency tensor* of  $G$ .

We first describe how to construct, for any  $S \subseteq V$  and  $i \in [r]$ , vectors  $u_{i,S}$  of length  $d'$  (defined below) such that for all *pairwise disjoint*  $S_1, \dots, S_k \subseteq V$ , we have:

$$u_{1,S_1} \cup \dots \cup u_{k,S_k} = [d'] \iff S_1 \cup \dots \cup S_k \text{ forms an } r\text{-hyperclique in } G$$

Define  $\mathcal{I} = \{(c_1, \dots, c_k) \mid \forall i, c_i \in \mathbb{N} \cup \{0\} \text{ and } \sum_i c_i = r\}$ . We define  $u_{i,S}$  with dimension  $d' = \binom{m}{r} \cdot |\mathcal{I}|$  as follows. For  $e = \{v_1, \dots, v_r\} \subseteq V$  and  $(c_1, \dots, c_k) \in \mathcal{I}$ ,

$$u_{i,S}[e, (c_1, \dots, c_k)] = \begin{cases} 0 & \text{if } c_i = 0 \\ 1 & \text{if } |e \cap S| \neq c_i \\ x_e & \text{otherwise} \end{cases}$$

**Correctness.** Let  $S_1, \dots, S_k \subseteq V$  be pairwise disjoint, and  $e = \{v_1, \dots, v_r\} \subseteq S_1 \cup \dots \cup S_k$  be chosen arbitrarily. Define  $\vec{\alpha} \in (\mathbb{N} \cup \{0\})^k$  as  $\alpha_i = |e \cap S_i|$ . Since the sets  $S_i$  are pairwise disjoint,  $\sum_i \alpha_i = |e| = r$ .

To see the forward direction, suppose  $e \notin E$ . i.e.,  $x_e = 0$ . Then for each  $i$  such that  $\alpha_i = 0$ , we have  $u_{i,S_i}[e, \vec{\alpha}] = 0$ . For those  $i$  where  $\alpha_i \neq 0$ , we know  $\alpha_i = |e \cap S_i|$  by definition, and thus  $u_{i,S_i}[e, \vec{\alpha}] = x_e = 0$ . Hence  $(u_{1,S_1} \cup \dots \cup u_{k,S_k})[i] = 0$ .

For the reverse direction, suppose  $S_1 \cup \dots \cup S_k$  indeed forms an  $r$ -hyperclique in  $G$ . Let  $e = \{v_1, \dots, v_r\} \subseteq S_1 \cup \dots \cup S_r$ , and  $\vec{c} := (c_1, \dots, c_r) \in \mathcal{I}$  be chosen arbitrarily. If  $c_i = \alpha_i$  for all  $i$ , then for any choice of  $j \in [k]$  such that  $\alpha_j \neq 0$ , we have  $u_{j,S_j}[e, \vec{c}] = x_e = 1$ . Else  $\vec{c} \neq \vec{\alpha}$ . Since  $\sum_i \alpha_i = \sum_i c_i = r$ , there must exist  $j \in [k]$  such that  $c_j \neq \alpha_j$  (equivalently  $c_j \neq |e \cap S_j|$ ) and  $c_j \neq 0$ . Then by definition, we have  $u_{j,S_j}[e, \vec{c}] = 1$ .

For  $e \notin S_1 \cup \dots \cup S_r$  and any choice of  $\vec{c} \in \mathcal{I}$ , there must be an  $i$  such that  $c_i \neq 0$  and  $|e \cap S_i| \neq c_i$  (since otherwise, we would have  $|e \cap (S_1 \cup \dots \cup S_r)| = r$ , a contradiction). This means that  $u_{i,S_i}[e, \vec{c}] = 1$  as in the previous paragraph.

### The reduction.

We construct a monotone circuit  $R$  that takes as input an  $r$ -uniform hypergraph  $G = (V, E)$  with  $|V| = m$  encoded as variables  $\{x_S \mid S \subseteq V \text{ and } |S| = r\}$  and produces as output  $k$  lists

$A_1, \dots, A_k$  of  $n = \binom{m}{\ell/k}$  Boolean vectors each of dimension  $d = \binom{k}{2} \cdot m + d'$  such that

$$\text{klnt}_{n,d}^*(A_1, \dots, A_k) = 1 \iff G \text{ has an } r\text{-hyperclique of size } \ell.$$

The idea is to use the fact that a set  $S \subseteq V$  forms an  $r$ -hyperclique of size  $\ell$  in  $G$  if and only if  $S$  can be partitioned into  $S_1, \dots, S_k$  each of size  $\ell/k$  such that  $S_1 \cup \dots \cup S_k$  is an  $r$ -hyperclique in  $G$ .

The vectors in  $A_i$  will be potential candidates for the set  $S_i$ . We will think of each such vector as having two parts. The first part will contain  $\binom{k}{2}$  blocks (each of  $m$  bits) that will be used to verify that  $S_1, \dots, S_k$  are pairwise disjoint. The second part uses the  $u_{i,S}$  vectors constructed earlier to check that  $S_1 \cup \dots \cup S_k$  forms an  $r$ -hyperclique.

We define the vectors in  $A_1, \dots, A_k$  as follows. The vector  $a_{i,S}$  is the vector  $u_{i,S}$  prefixed by  $\binom{k}{2}$  blocks each of  $m$  bits. These are indexed by  $\{j_1, j_2\} \in \binom{[k]}{2}$  and defined as:

$$a_{i,S}[\{j_1, j_2\}] = \begin{cases} \chi_{\overline{S}} & \text{if } i \in \{j_1, j_2\} \\ 0 & \text{otherwise.} \end{cases}$$

where  $\chi_{\overline{S}}$  is the characteristic vector of  $\overline{S}$ . As mentioned earlier, the above  $\binom{k}{2}m$  bits are followed by the vector  $u_{i,S}$ . This completes the definition of  $a_{i,S}$ .

**Correctness.** We show that  $G$  contains an  $r$ -hyperclique of size  $\ell$  if and only if  $(A_1, \dots, A_k)$  is a 1-instance of  $\text{klnt}_{n,d}^*$ .

“ $\implies$ ”: Suppose  $G$  contains an  $r$ -hyperclique of size  $\ell$  on  $T \subseteq V$ . Let  $T_1, \dots, T_k \subseteq T$  be an arbitrary partition of  $T$  into  $k$  parts each of size  $\ell/k$ . We claim that  $a_{1,T_1} \cup \dots \cup a_{k,T_k} = [d]$ .

To see that the bits in the first  $\binom{k}{2}$  blocks are all 1s, choose any index  $i$  in this range. The index  $i$  is in a block  $\{j_1, j_2\} \in \binom{[k]}{2}$ . Then exactly two of the vectors among  $a_{1,T_1}, \dots, a_{k,T_k}$  are non-zero in the block  $\{j_1, j_2\}$ , namely  $a_{j_1,T_{j_1}}$  and  $a_{j_2,T_{j_2}}$ . Recall that the  $T_i$  are pairwise disjoint. In particular,  $T_{j_1}$  and  $T_{j_2}$  are disjoint. Then by definition  $a_{j_1,T_{j_1}}[\{j_1, j_2\}] = \chi_{\overline{T_{j_1}}}$  and  $a_{j_2,T_{j_2}}[\{j_1, j_2\}] = \chi_{\overline{T_{j_2}}}$ . Hence  $a_{j_1,T_{j_1}}[\{j_1, j_2\}] \cup a_{j_2,T_{j_2}}[\{j_1, j_2\}] = \overline{1}$  which implies  $(a_{1,T_1} \cup \dots \cup a_{k,T_k})[i] = 1$ . Since the choice of  $i$  was arbitrary, the first part of  $a_{1,T_1} \cup \dots \cup a_{k,T_k}$  is all 1s.

The second part of each  $a_{i,S_i}$  is simply  $u_{i,S_i}$ . Since the  $T_i$  are pairwise disjoint and  $T = T_1 \cup \dots \cup T_k$  is an  $r$ -hyperclique, we have  $u_{1,S_1} \cup \dots \cup u_{k,S_k} = [d']$ . Thus,  $a_{1,T_1} \cup \dots \cup a_{k,T_k} = [d]$ . “ $\impliedby$ ”: Suppose  $(A_1, \dots, A_k)$  is a 1-instance of  $\text{klnt}_{n,d}^*$ . Then there exists  $a_{1,S_1}, \dots, a_{k,S_k}$  such that  $a_{1,S_1} \cup \dots \cup a_{k,S_k} = [d]$ . We will show that  $S_1 \cup \dots \cup S_k$  forms an  $r$ -hyperclique of size  $\ell$  in  $G$ .

Observe that in the first  $m \cdot \binom{k}{2}$  bits, for any block  $\{j_1, j_2\} \in \binom{[k]}{2}$ , exactly two of the vectors among  $a_{1,S_1}, \dots, a_{k,S_k}$  are non-zero namely  $a_{j_1,S_{j_1}}$  and  $a_{j_2,S_{j_2}}$ . Since  $a_{j_1,S_{j_1}}[\{j_1, j_2\}] \cup a_{j_2,S_{j_2}}[\{j_1, j_2\}] = \overline{1}$ , we have  $\chi_{\overline{S_1}} \cup \chi_{\overline{S_2}} = \overline{1}$ . This means  $S_{j_1}$  and  $S_{j_2}$  are disjoint. This is true for every choice of  $\{j_1, j_2\}$ , and hence all of  $S_1, \dots, S_k$  must be pairwise disjoint.

Since  $a_{1,S_1} \cup \dots \cup a_{k,S_k} = [d]$ , and the second part of each  $a_{i,S_i}$  equals  $u_{i,S_i}$  by definition, it must be the case that  $u_{1,S_1} \cup \dots \cup u_{k,S_k} = [d']$ . Combining this with the fact that the  $S_i$  are pairwise disjoint, we can conclude that  $S_1 \cup \dots \cup S_k$  forms an  $r$ -hyperclique in  $G$ .

**Analysis.** Finally, we argue about the complexity of the reduction. Note that  $u_{i,S}$  have dimension  $d'$ , where  $d' = \binom{m}{r}|\mathcal{I}| = \binom{m}{r} \cdot \binom{r+k-1}{k-1}$  and thus  $d = \binom{k}{2} \cdot m + \binom{m}{r} \cdot \binom{r+k-1}{k-1}$ , which is the dimension of  $a_{i,S}$  vectors.

Furthermore, each coordinate of  $u_{i,S}$  can be computed as a simple monotone projection (i.e. either a constant or a single variable) from the input graph. The numbers of such projections is  $d'$ . Thus, overall, the  $u_{i,S}$  vectors can be computed by a circuit of size  $O(m^r)$

(recall that  $k, r$  are constants). The first part of  $a_{i,S}$  is also a projection and thus can be computed with size  $O(m)$ . Thus, the overall size of the projection is  $O(\binom{m}{\ell/k} \cdot (m + m^r))$ .

Thus, if  $\text{Int}_{n,d}$  has size  $s$  monotone (resp. depth- $h$ ) circuit, then we get a monotone (resp. depth- $h$ ) circuit for  $\text{HypClique}_{r,\ell}^m$  of size  $s + O(\binom{m}{\ell/k} \cdot (m + m^r))$ .

By our assumed hypothesis on the circuit size of  $\text{HypClique}_{r,\ell}^m$ , we get the following bounds.

$$\begin{aligned}
s &\geq \binom{m}{\ell}^{1-\varepsilon} - O\left(\binom{m}{\ell/k} \cdot (m + m^r)\right) \geq \left(\frac{m}{\ell}\right)^{(1-\varepsilon)\ell} - O\left(\binom{m}{\ell/k} \cdot m^r\right) \\
&= \left(\frac{me}{\ell/k}\right)^{k \cdot \frac{\ell}{k} \cdot (1-\varepsilon)} \cdot \frac{1}{(ke)^{\ell(1-\varepsilon)}} - O\left(\binom{m}{\ell/k} \cdot m^r\right) \\
&\geq \left(\frac{m}{\ell/k}\right)^{k(1-\varepsilon)} \cdot \frac{1}{2^{O(\ell)}} - O\left(\binom{m}{\ell/k} \cdot m^r\right) \\
&\geq \left(\frac{m}{\ell/k}\right)^{k(1-2\varepsilon)} = n^{k(1-2\varepsilon)}
\end{aligned}$$

where the final inequality uses the fact that  $\ell$  is large enough,  $\ell = o(m)$ , and constant  $r$ , which implies that for any small enough constant  $\varepsilon$  and large enough  $m$

$$2^{O(\ell)} \leq \left(\frac{m}{\ell/k}\right)^{o(1)} \quad \text{and} \quad \binom{m}{\ell/k} \cdot m^r \leq \left(\frac{m}{\ell/k}\right)^{k \cdot (1-2\varepsilon)}$$

This concludes the proof. ◀

► **Remark 16 (Shortcomings of this proof framework).** While Theorem 11 yields a simple proof of the monotone hardness of  $\text{Int}_{n,d}$ , we note that it only holds in the regime of moderately large  $d$  (i.e.  $d = \exp((\log n)^{\Omega(1)})$ ). This is due to the fact that we need a near-tight lower bound (i.e.  $\binom{m}{\ell}^{1-\varepsilon}$ ) for  $\ell$ -clique. We only have such lower bounds in the regime of small  $\ell = o(\log m)$ . While we have strong lower bounds for larger values of  $\ell$  [5, 16, 12], they stop being near-tight in this sense.

Though the  $\text{Int}_{n,d}$  problem is already interesting in this regime of parameters [2], the importance of  $\text{Int}_{n,d}$  for many fine-grained problems (e.g. SETH) lies in the setting of small  $d$ . In this setting, even assuming the best possible lower bounds for clique, e.g. a near-tight lower bound in the entire regime where  $\ell = o(m)$ ,<sup>8</sup> this would only yield a lower bound for  $\text{Int}_{n,d}$  in the regime where  $d = \Omega((\log n)^2)$ .

## 4 Monotone version of OV Conjecture

In this section, we prove Theorem 5.

### Distributions for 0-inputs and 1-inputs

We define a distribution  $\mathbf{D}_0$  over 0-inputs of  $\text{Int}_{n,d}$  as follows:

1. Choose  $i, j \in [n]$  uniformly and independently at random.

<sup>8</sup> Current known lower bounds [12, 20] only work for  $\ell \leq \sqrt{m}$ , and are furthermore not tight enough for the above reduction to be applicable when  $\ell = \omega(\log m)$ .

2. Choose a vector  $b \in \{0, 1\}^d$  by choosing  $b_1, \dots, b_d \in \{0, 1\}$  uniformly and independently at random.
3. Set  $u_i$  to  $b$  and  $v_j$  to  $\bar{b}$ .
4. For each  $k \neq i$ , set vector  $u_k$  to all 1s. For each  $k' \neq j$ , set vector  $v_{k'}$  to all 1s.

Observe that setting  $u_i = b$  and  $v_j = \bar{b}$  makes them orthogonal, and thus every input drawn from  $\mathbf{D}_0$  is a 0-input of  $\text{Int}$ .

We define the distribution  $\mathbf{D}_1$  as  $\mu_p$ , the distribution where each bit is set to 1 independently at random with probability  $p$ . We observe that inputs drawn from  $\mathbf{D}_1$  are 1-inputs of  $\text{Int}$  with very high probability:

► **Lemma 17.**

$$\Pr_{(u,v) \sim \mathbf{D}_1} [\text{Int}_{n,d}(u, v) = 1] \geq 1 - n^2 e^{-p^2 d}$$

**Proof.** Let  $x, y$  be  $d$ -bit strings drawn from the distribution  $\mu_p$ . The probability that  $x$  and  $y$  are disjoint is  $(1 - p^2)^d$ . By definition,  $\text{Int}(u, v) = 0$  if and only if there exists at least one pair  $u \in \mathcal{U}$  and  $v \in \mathcal{V}$  that are disjoint. Using a union bound over all  $n^2$  possible pairs of vectors, we get  $\Pr[\text{Int}(u, v) = 0] \leq n^2(1 - p^2)^d \leq n^2 e^{-p^2 d}$ . ◀

**Proof Outline.** As described in Section 1.1, the idea behind the approximation method which we use to prove Theorem 5 is to show that any small monotone circuit  $C$  defined on the inputs of  $\text{OV}_{n,d}$  can be approximated, w.r.t. the two distributions  $\mathbf{D}_0$  and  $\mathbf{D}_1$  defined above, by a monotone  $F$  from a well-defined family of ‘simple’ monotone functions that we will call *approximators*. Once this is done, we can easily show that  $C$  cannot have been computing  $\text{OV}_{n,d}$  since no approximator can even approximately compute  $\text{OV}_{n,d}$  w.r.t.  $\mathbf{D}_0$  and  $\mathbf{D}_1$ . The hard part is to prove that the circuit  $C$  indeed has an approximator with low error. This is done carefully at each gate of the circuit, replacing it with a suitable approximator in a way that does not increase the error too much.

The conceptually new part of the lower bound of this paper is the definition of the approximator as more ‘standard-issue’ constructions do not seem to work (see Remark 18 below). Showing that this new kind of approximator is amenable to the inductive argument requires some technical work, with the highlight being a simpler version of the Kruskal-Katona theorem (Lemma 22 below) that allows us to bound the error at OR gates effectively.

► **Remark 18.** In this technical remark, we sketch why the approximation method applied with more ‘standard’ approximators do not seem to work in our setting here. We will assume knowledge of the standard applications of the approximation method.

Note that the marginal probabilities in  $\mathbf{D}_0$  are  $1/2$  (and cannot be higher for a similar construction) conditioned on a vector  $u_i$  being chosen in Step 1 of the sampling procedure. This technical point prevents us from using ‘standard’ techniques such as CNF approximators (which would otherwise be very natural) as in e.g. [15, Section 3] in what follows. To apply these ideas, we would need to be able to trade off the probability that a clause of width  $w + 1$  is set to 0 under  $\mathbf{D}_0$  (roughly  $n^{-1}2^{-(w+1)}$ ) with the number of clauses required to make a sunflower under  $\mathbf{D}_1$ , which is roughly  $w^w$  using the best known robust sunflower lemmas [30]. Unfortunately, this means that removing all clauses of width more than  $w$  incurs an error bound of up to  $w^w \cdot n^{-1}2^{-O(w)}$  which we need to be bound by  $n^{-(2-\epsilon)}$  for a union bound over the gates of the circuit. Unfortunately, there is no choice of  $w$  that satisfies this constraint. This renders the standard technique of using such approximators inadmissible in this context.

We now begin the main proof.

Recall that the inputs to  $\text{Int}_{n,d}$  are two  $n$ -tuples  $\mathcal{U}$  and  $\mathcal{V}$  of  $d$ -dimensional vectors. In this context, we call a function  $B : \{0, 1\}^d \rightarrow \{0, 1\}$  a ‘local function’ if the inputs to  $B$  are a vector  $u \in \mathcal{U}$  or a vector  $v \in \mathcal{V}$ . i.e., the function  $B$  does not take inputs from more than one vector.

We are now ready to describe the structure of our *approximator*.

► **Definition 19.** Let  $0 \leq m \leq n$  and  $0 \leq \delta \leq 1$  be any fixed parameters. A Boolean function  $\tilde{F} : (\{0, 1\}^d)^n \times (\{0, 1\}^d)^n \rightarrow \{0, 1\}$  is an  $(m, \delta)$ -approximator if it is a constant, or it can be written as

$$\tilde{F} = \bigwedge_{i \in S_{\mathcal{U}}} B_i \wedge \bigwedge_{j \in S_{\mathcal{V}}} C_j$$

where  $S_{\mathcal{U}}, S_{\mathcal{V}} \subseteq [n]$  and the following three properties are satisfied:

1. For each  $i \in S_{\mathcal{U}}$  ( $j \in S_{\mathcal{V}}$ ), the function  $B_i$  ( $C_j$ ) is a local function on  $u_i$  ( $v_j$  respectively).
2.  $|S_{\mathcal{U}}| \leq m$  and  $|S_{\mathcal{V}}| \leq m$ .
3. For all  $i \in S_{\mathcal{U}}$ , and all  $j \in S_{\mathcal{V}}$ , we have:

$$\Pr_{x \sim \mathbf{D}_1} [B_i(x) = 0] \geq \delta \text{ and } \Pr_{x \sim \mathbf{D}_1} [C_j(x) = 0] \geq \delta$$

### Constructing an $(m, \delta)$ -approximator

We begin by showing how to construct an  $(m, \delta)$ -approximator for a monotone circuit for any  $\delta < (1 - p)$  inductively starting at the leaves. We denote the approximator at gate  $g$  with  $\tilde{F}_g$ .

- **Leaves:** The leaves of the circuit are labelled by either variables or constants. For a variable  $x_i$ , the approximator is simply the function  $x_i$ . It is straightforward to see that properties 1 and 2 are satisfied in this case. To see property 3, observe that the probability over  $\mathbf{D}_1$  that  $x_i$  is 1 is  $p$ . Since  $\delta < (1 - p)$ , property 3 is satisfied. For a leaf that is labelled by a constant 0 or 1, the approximator is also the same constant.
- **$g \wedge h$ :**  
Let  $\tilde{F}_g$  and  $\tilde{F}_h$  be  $(m, \delta)$ -approximators for  $g$  and  $h$  respectively. It is easy to see that  $\tilde{F}_g \wedge \tilde{F}_h$  satisfies properties 1 and 3 as conjunctions of local functions are also local functions, and for any two functions that satisfy property 3, their conjunction will also satisfy property 3. If  $\tilde{F}_g \wedge \tilde{F}_h$  also satisfies property 2, then we define  $\tilde{F}_{g \wedge h} := \tilde{F}_g \wedge \tilde{F}_h$ . Otherwise define  $\tilde{F}_{g \wedge h} := 0$ .
- **$g \vee h$ :**  
Let  $\tilde{F}_g = \bigwedge_{i \in S_{\mathcal{U}}^g} B_i^g \wedge \bigwedge_{j \in S_{\mathcal{V}}^g} C_j^g$  and  $\tilde{F}_h = \bigwedge_{i \in S_{\mathcal{U}}^h} B_i^h \wedge \bigwedge_{j \in S_{\mathcal{V}}^h} C_j^h$  be  $(m, \delta)$ -approximators for  $g$  and  $h$  respectively. Then we have:

$$\begin{aligned} \tilde{F}_g \vee \tilde{F}_h &= \bigwedge_{i \in S_{\mathcal{U}}^g, i' \in S_{\mathcal{U}}^h} B_i^g \vee B_{i'}^h \wedge \bigwedge_{i \in S_{\mathcal{U}}^g, j' \in S_{\mathcal{V}}^h} B_i^g \vee C_{j'}^h \\ &\quad \wedge \bigwedge_{j \in S_{\mathcal{V}}^g, i' \in S_{\mathcal{U}}^h} C_j^g \vee B_{i'}^h \wedge \bigwedge_{j \in S_{\mathcal{V}}^g, j' \in S_{\mathcal{V}}^h} C_j^g \vee C_{j'}^h \end{aligned} \quad (1)$$

From the above, we include in our approximator only those functions that are local functions and satisfy property 3. This is done as follows. Let  $T_{\mathcal{U}} = \{i \in S_{\mathcal{U}}^g \cap S_{\mathcal{U}}^h \mid (B_i^g \vee$

$B_i^h$ ) satisfies property 3} and  $T_V = \{j \in S_V^g \cap S_V^h \mid (C_j^g \vee C_j^h) \text{ satisfies property 3}\}$ . If  $|T_U| = |T_V| = 0$ , then we define  $\tilde{F}_{g \vee h} = 1$ . Else we define  $\tilde{F}_{g \vee h}$  as

$$\tilde{F}_{g \vee h} = \bigwedge_{i \in T_U} (B_i^g \vee B_i^h) \wedge \bigwedge_{j \in T_V} (C_j^g \vee C_j^h)$$

Observe that since each function in this conjunction is a local function,  $\tilde{F}_{g \vee h}$  satisfies property 1. Also note that  $|T_U| \leq |S_U^g \cap S_U^h| \leq |S_U^g| \leq m$ , and similarly  $|T_V| \leq |S_V^g \cap S_V^h| \leq |S_V^g| \leq m$ . So,  $\tilde{F}_{g \vee h}$  satisfies property 2 as well. Finally, property 3 is satisfied by  $\tilde{F}_{g \vee h}$  due to the definitions of  $T_U$  and  $T_V$ .

## Error bounds

We first show that any  $(m, \delta)$ -approximator has large error on our distributions.

► **Lemma 20.** *Let  $\tilde{F} = \bigwedge_{i \in S_U} f_i \wedge \bigwedge_{j \in S_V} g_j$  be any  $(m, \delta)$ -approximator. Then, either  $\tilde{F}$  is the constant 0, or*

$$\Pr_{\mathbf{D}_0}[\tilde{F} = 0] \leq \frac{2m}{n}$$

**Proof.** Suppose  $\tilde{F}$  is not the constant 0. Assume, without loss of generality, that  $|S_U| > 0$ . For some  $i \in S_U$ , let  $B := f_i$  be a local function of  $u_i$ . Since  $B$  is monotone and not the constant 0, it must be that  $B(1^d) = 1$ . Recall that all vectors not chosen by  $\mathbf{D}_0$  in step 1 are assigned  $1^d$ . The probability that  $\mathbf{D}_0$  picks  $u_i$  in step 1 is  $1/n$ . Thus,  $\Pr_{(u,v) \sim D_0}[B(u_i) = 0] \leq 1/n$ . By a union bound over all the local functions in  $\tilde{F}$ , the probability over  $\mathbf{D}_0$  that at least one of them is 0 is at most  $2m/n$ . ◀

We now show that our construction of the  $(m, \delta)$ -approximator incurs very small error at each gate. An observation that is trivial from our construction is that at AND gates (OR gates), our approximator does not incur any new errors on inputs drawn from  $\mathbf{D}_0$  ( $\mathbf{D}_1$  respectively). So, we only need to bound the  $\mathbf{D}_1$ -error at AND gates, and  $\mathbf{D}_0$  error at OR gates.

► **Lemma 21 (Errors at  $\wedge$  gates).** *Let  $\tilde{F}_g$  and  $\tilde{F}_h$  be  $(m, \delta)$ -approximators for the functions computed at gates  $g$  and  $h$  respectively. Then we have:*

$$\Pr_{\mathbf{D}_1}[\tilde{F}_g \wedge \tilde{F}_h \neq \tilde{F}_{g \wedge h}] \leq e^{-m\delta}$$

**Proof.** Let  $\tilde{F}_g = \bigwedge_{i \in S_U^g} B_i^g \wedge \bigwedge_{j \in S_V^g} C_j^g$  and  $\tilde{F}_h = \bigwedge_{i \in S_U^h} B_i^h \wedge \bigwedge_{j \in S_V^h} C_j^h$ . Recall that  $\tilde{F}_g \wedge \tilde{F}_h$  always satisfies properties 1 and 3. If property 2 was also satisfied, we defined  $\tilde{F}_{g \wedge h} = \tilde{F}_g \wedge \tilde{F}_h$ . Thus, in this case, the lemma is trivially true.

When  $\tilde{F}_g \wedge \tilde{F}_h$  violates property 2, we defined  $\tilde{F}_{g \wedge h}$  to be the constant 0. In this case it suffices to determine the probability that  $\tilde{F}_g \wedge \tilde{F}_h$  is 1. We can write  $\tilde{F}_g \wedge \tilde{F}_h = \bigwedge_{i \in S_U^g \cup S_U^h} P_i \wedge \bigwedge_{j \in S_V^g \cup S_V^h} Q_j$  where  $P_i$  and  $Q_j$  are defined as follows.

$$P_i = \begin{cases} B_i^g \wedge B_i^h & \text{for } i \in S_U^g \cap S_U^h \\ B_i^g & \text{for } i \in S_U^g \setminus S_U^h \\ B_i^h & \text{for } i \in S_U^h \setminus S_U^g \end{cases} \quad Q_j = \begin{cases} C_j^g \wedge C_j^h & \text{for } j \in S_V^g \cap S_V^h \\ C_j^g & \text{for } j \in S_V^g \setminus S_V^h \\ C_j^h & \text{for } j \in S_V^h \setminus S_V^g \end{cases}$$

Since property 2 is violated, assume without loss of generality that  $|S_{\mathcal{U}}^g \cup S_{\mathcal{U}}^h| > m$ . Then we have,

$$\begin{aligned} \Pr_{\mathbf{D}_1} \left[ \bigwedge_{i \in S_{\mathcal{U}}^g \cup S_{\mathcal{U}}^h} P_i = 1 \right] &= \prod_i \Pr[P_i = 1] \\ &\leq (1 - \delta)^m \\ &\leq e^{-m\delta} \end{aligned}$$

where the first equality is because the  $P_i$ s are on a disjoint sets of variables, and thus the events that they are 0 are independent. The inequality in the next line is from property 3.  $\blacktriangleleft$

We need the following technical lemma before bounding the errors incurred at OR gates.

► **Lemma 22.** *Let  $B : \{0, 1\}^d \rightarrow \{0, 1\}$  be any monotone Boolean function. Let  $0 < p < q < 1$  and let  $\mu_p$  and  $\mu_q$  be probability distributions where each bit is set to 1 with probability  $p$  and  $q$  respectively. Then,*

$$\Pr_{u \sim \mu_q} [B(u) = 0] \leq \left( \Pr_{u \sim \mu_p} [B(u) = 0] \right)^{\lfloor q/p \rfloor}$$

**Proof.** Let  $t = \lfloor q/p \rfloor$  and let  $u_1, u_2, \dots, u_t \sim \mu_p$  be  $d$ -bit vectors drawn independently at random from  $\mu_p$ . Let  $v = u_1 \vee u_2 \vee \dots \vee u_t$  be the bit-wise OR of the  $u_i$ s. We denote the  $i$ th bit of vector  $v$  with  $v_i$  and let  $q'$  denote  $\Pr[v_i = 1]$ . We will first observe that  $q' \leq q$  and hence the expected number of zeros in a string drawn from the distribution  $\mu_{q'}$  will be greater than equal to the expected number of zeros in a string drawn from the distribution  $\mu_q$ . Using the definition of  $v$  and a union bound, we can conclude that  $q' \leq q$ :

$$q' = \Pr_{v \sim \mu_{q'}} [v_i = 1] = \Pr_{u_1, u_2, \dots, u_t \sim \mu_p} [(u_{1i} = 1) \vee (u_{2i} = 1) \vee \dots \vee (u_{ti} = 1)] \leq tp \leq q$$

Since  $B$  is monotone,  $B$  is more likely to output 0 on a string drawn from  $\mu_{q'}$  than on a string drawn from the distribution  $\mu_q$ . i.e.,  $\Pr_{v \sim \mu_q} [B(v) = 0] \leq \Pr_{v \sim \mu_{q'}} [B(v) = 0] = \Pr_{u_1, u_2, \dots, u_t \sim \mu_p} [B(u_1 \vee u_2 \vee \dots \vee u_t) = 0]$ . Combining this with the fact if  $B(u_1 \vee u_2 \vee \dots \vee u_t) = 0$  then  $\forall i \in [t], B(u_i) = 0$ , we can conclude:

$$\begin{aligned} \Pr_{u \sim \mu_q} [B(u) = 0] &\leq \Pr_{u_1, u_2, \dots, u_t \sim \mu_p} [B(u_1 \vee u_2 \vee \dots \vee u_t) = 0] \leq \prod_{i=1}^t \Pr_{u_i \sim \mu_p} [B(u_i) = 0] \\ &= \left( \Pr_{u_i \sim \mu_p} [B(u_i) = 0] \right)^{\lfloor q/p \rfloor} \end{aligned} \tag{2}$$

Recall that  $p$  is the probability with which  $\mathbf{D}_1$  sets each bit to 1. We now determine the new errors incurred at  $\vee$  gates.  $\blacktriangleleft$

► **Lemma 23 (Errors at  $\vee$  gates).** *Fix parameters  $0 \leq m \leq n$  and  $0 \leq \delta \leq 1$  such that  $\delta^{\lfloor 1/2p \rfloor} \leq 1/n$ . Let  $\tilde{F}_g$  and  $\tilde{F}_h$  be  $(m, \delta)$ -approximators for the functions computed at gates  $g$  and  $h$  respectively. Then we have:*

$$\Pr_{D_0} [\tilde{F}_g \vee \tilde{F}_h \neq \tilde{F}_{g \vee h}] \in O \left( \frac{m^2}{n^2} \right)$$

**Proof.** Let  $\tilde{F}_g = \bigwedge_{i \in S_{\mathcal{U}}^g} B_i^g \wedge \bigwedge_{j \in S_{\mathcal{V}}^g} C_j^g$  and  $\tilde{F}_h = \bigwedge_{i \in S_{\mathcal{U}}^h} B_i^h \wedge \bigwedge_{j \in S_{\mathcal{V}}^h} C_j^h$  be  $(m, \delta)$ -approximators for  $g$  and  $h$  respectively. Recall that  $\tilde{F}_g \vee \tilde{F}_h$  looks like equation 1. When constructing  $\tilde{F}_{g \vee h}$ , we removed functions of two kinds from equation 1:

1. Functions that were not local functions. These come in two types:
  - a.  $(B_i^g \vee B_j^h)$  for some  $i \neq j$ . Recall that in the random process that defines  $\mathbf{D}_0$ , all vectors in  $\mathcal{U}$  except one (randomly chosen) vector are made  $1^d$ . Thus, it is always true that either  $u_i$  or  $u_j$  is assigned  $1^d$ . This means either  $B_i^g$  or  $B_j^h$  must be 1 on every input drawn from  $\mathbf{D}_0$ . Hence there is no error when not including such functions in  $\tilde{F}_{g \vee h}$ .
  - b.  $(B_i^g \vee C_j^h)$ . If either of  $i$  or  $j$  were not the chosen indices in step 1 of  $D_0$ , then similar to the previous case, either  $B_i^g$  or  $C_j^h$  will be 1. The probability that  $i$  as well as  $j$  were the chosen indices in step 1 of  $\mathbf{D}_0$  is  $1/n^2$ . There are at most  $m^2$  such functions, and hence by a union bound, the error incurred is at most  $m^2/n^2$ .
2. Functions that were local functions, but did not satisfy property 3. Suppose  $B_i^g \vee B_i^h$  did not satisfy property 3. We will determine the probability that  $B_i^g \vee B_i^h$  equals 0. Observe that if  $i$  was the first index chosen by the distribution  $\mathbf{D}_0$  in step 1, then each bit of the vector  $u_i$  is chosen uniformly and independently at random. Let  $B = B_i^g \vee B_i^h$ . Since  $B$  does not satisfy property 3, we have  $\Pr_{u_i \sim \mathbf{D}_1}[B(u_i) = 0] < \delta$ . Applying lemma 22 to  $B$  with  $q = 1/2$ , we obtain:

$$\Pr_{u_i \sim \mu_{1/2}}[B(u_i) = 0] \leq \left( \Pr_{x \sim \mathbf{D}_1}[B(u_i) = 0] \right)^{\lfloor 1/2p \rfloor} < \delta^{\lfloor 1/2p \rfloor}$$

Thus,  $\Pr_{(u_i, \nu) \sim \mathbf{D}_0}[B(u_i) = 0] = \Pr[i \text{ was chosen}] \cdot \Pr_{u_i \sim \mu_{1/2}}[B(u_i) = 0] < \delta^{\lfloor 1/2p \rfloor} / n$ . By a union bound over at most  $m^2$  such functions, the total error incurred is at most  $\delta^{\lfloor 1/2p \rfloor} m^2 / n$ . Since  $\delta^{\lfloor 1/2p \rfloor} \leq 1/n$ , the lemma follows. ◀

We can now show the main theorem by combining the error bounds established above.

► **Theorem 5.** *For every  $0 < \epsilon < 1$ , there exists a constant  $c$  such that for all  $d \geq c \log n$  any monotone circuit computing  $\text{Int}_{n,d}$  requires at least  $\Omega(n^{2-\epsilon})$  many gates.*

**Proof.** Set  $p = \epsilon/4$ ,  $c = 100/\epsilon^2$ ,  $d = c \log n$ ,  $m = n^\epsilon$  and  $\delta = \frac{1}{n^{\epsilon/2}}$ . Let  $C$  be any monotone circuit computing  $\text{Int}_{n,d}$  using  $s$  many gates. Let  $\tilde{F}$  be the  $(m, \delta)$ -approximator for  $C$  constructed as described in Section 4. As mentioned before, the approximation at AND gates incurs new errors only over the distribution  $\mathbf{D}_1$ . Similarly, when approximating OR gates, we only incur new errors on inputs drawn from  $\mathbf{D}_0$ . Using error bounds from Lemma 23, Lemma 21, and a union bound over the  $s$  gates of  $C$ , we have:

$$\Pr_{\mathbf{D}_0}[\tilde{F} \neq C] \leq s \cdot O\left(\frac{m^2}{n^2}\right) \tag{3}$$

$$\Pr_{\mathbf{D}_1}[\tilde{F} \neq C] \leq s\epsilon^{-m\delta} \tag{4}$$

Now we have two cases with respect to whether  $\tilde{F}$  is the constant 0 or a non-constant function.

- **Case 1:**  $\tilde{F}$  is the constant 0. In this case,  $\tilde{F}$  is wrong on all 1-inputs of  $\text{Int}$  drawn from the distribution  $\mathbf{D}_1$ . So, we have:

$$\begin{aligned} \Pr_{\mathcal{U}, \mathcal{V} \sim \mathbf{D}_1} [C(\mathcal{U}, \mathcal{V}) \neq \tilde{F}(\mathcal{U}, \mathcal{V})] &= \Pr_{\mathcal{U}, \mathcal{V} \sim \mathbf{D}_1} [C(\mathcal{U}, \mathcal{V}) = 1 \text{ and } \tilde{F}(\mathcal{U}, \mathcal{V}) = 0] \\ &= \Pr_{\mathcal{U}, \mathcal{V} \sim \mathbf{D}_1} [\text{Int}_{n,d}(\mathcal{U}, \mathcal{V}) = 1] \\ &\geq 1 - n^2 e^{-p^2 d} \in \Omega(1) \end{aligned}$$

where the first two equalities are because  $\tilde{F}$  is assumed to be 0. The final inequality is from Lemma 17. Combining this with Equation 4, we obtain:

$$\Omega(1) \leq \Pr_{\mathbf{D}_1} [C \neq \tilde{F}] \leq s e^{-m\delta}.$$

This gives a lower bound of  $s = \Omega(e^{m\delta})$ .

- **Case 2:**  $\tilde{F}$  accepts at least one input. Combining the error bound from Lemma 20 with Equation 3, we get:

$$1 - \frac{2m}{n} \leq \Pr_{\mathcal{U}, \mathcal{V} \sim \mathbf{D}_0} [C(\mathcal{U}, \mathcal{V}) \neq \tilde{F}(\mathcal{U}, \mathcal{V})] \leq s \cdot O\left(\frac{m^2}{n^2}\right)$$

This allows us to conclude  $s \geq \Omega\left(\frac{n^2}{m^2}\right) \left(1 - \frac{2m}{n}\right) = \Omega(n^{2(1-\epsilon)})$ .

Taking into account both of the above two cases, we get  $s \geq \min(\Omega(n^{2(1-\epsilon)}), e^{n^{\epsilon/2}})$ . Thus, for large  $n$ , we have  $s \geq \Omega(n^{2(1-\epsilon)})$ . ◀

## 5 Formula and Branching Program Lower Bounds

In this section we show lower bounds on the size of (not necessarily monotone) formulas and branching programs computing  $\text{Int}_{n,d}$ . We use the classic technique of Nechiporuk [25, 29] that relates the formula and branching program complexity of a function with the number of sub-functions with respect to disjoint subsets of variables.

Throughout this section, we shall use  $L(f)$  to denote the number of leaves in a smallest Boolean formula (with arbitrary binary gates)<sup>9</sup> computing  $f$ , and  $\text{BP}(f)$  to denote the number of nodes in a smallest branching program computing  $f$ . We first state Nechiporuk's theorem.

► **Theorem 24** ([25, 29]). *Let  $f$  be a Boolean function on a set of variables  $X$ . Let  $Y_1, Y_2, \dots, Y_m$  be disjoint subsets of  $X$ , and let  $s_i$  be the number of distinct sub-functions of  $f$  on  $Y_i$ . Then, there exists a constant  $\epsilon > 0$  such that*

$$L(f) \geq \frac{1}{4} \sum_{i=1}^m \log s_i \tag{5}$$

$$\text{BP}(f) \geq \epsilon \sum_{i=1}^m \frac{\log s_i}{\log \log s_i} \tag{6}$$

► **Theorem 25.** *We obtain lower bounds in two separate ranges of  $d$ :*

<sup>9</sup> A similar proof also works for the case where the fan-in of the formulas are bounded by some absolute constant  $c_0 \geq 2$  and each gate is labelled by an arbitrary Boolean function on at most  $c_0$  variables.

1. For  $d \geq c \log n$  for any  $c > 1$ ,

$$L(\text{Int}_{n,d}) \geq \Omega(n^2 d)$$

$$BP(\text{Int}_{n,d}) \geq \Omega\left(\frac{n^2 d}{\log(nd)}\right)$$

2. For  $d \leq \log n$ ,

$$L(\text{Int}_{n,d}) \geq \Omega(n 2^d / \sqrt{d})$$

$$BP(\text{Int}_{n,d}) \geq \Omega\left(\frac{n 2^d}{\sqrt{d} \log(2nd)}\right)$$

To use Theorem 24, we first bound the number of distinct sub-functions of  $\text{Int}_{n,d}$  with respect to the variables in any single vector.

► **Lemma 26.** *Let  $d \geq c \log n$  for any  $c > 1$ . Consider an instance of  $\text{Int}(\mathcal{A}, \mathcal{B})$ . Let  $a_i$  denote the  $i$ th vector in  $\mathcal{A}$  and  $b_i$  denote the  $i$ th vector in  $\mathcal{B}$ . Let  $s_i$  ( $t_i$ ) denote the number of distinct sub-functions of  $\text{Int}(\mathcal{A}, \mathcal{B})$  with respect to  $a_i$  ( $b_i$  respectively). Then the following inequalities hold:*

$$\left(\frac{2^{d-1}}{n\sqrt{d}}\right)^n \leq s_i, t_i \leq 2^{2nd}$$

**Proof.** We will show the statement for the number of distinct sub-functions  $s_1$  of  $\text{Int}$  with respect to  $a_1$ . A similar argument holds for each  $s_i$  and  $t_i$ .

Suppose we set all the vectors in  $\mathcal{A}$  other than  $a_1$  to  $\vec{1}$ , and set the vectors in  $\mathcal{B}$  to non-zero vectors. Then each of  $a_2, \dots, a_n$  will trivially intersect each vector in  $\mathcal{B}$  leaving us with the following sub-function on  $a_1$ :  $\bigwedge_{i=1}^n \bigvee_{j=1}^d (a_{1j} \wedge b_{ij})$ . Thus, setting  $b_1, \dots, b_n$  to non-zero vectors results in a monotone CNF on the variables of  $a_1$ .

**Lower bound.** Let  $\mathcal{H}_{d/2}$  be the set of all  $d$ -dim Boolean vectors having exactly  $d/2$  many ones. We claim that if we set each  $b_i \in \mathcal{B}$  to a distinct vector from  $\mathcal{H}_{d/2}$ , then each way of setting the  $b_i$ s results in distinct sub-functions. Observe that if the  $b_i$ s have exactly  $d/2$  many 1s, then each clause in the CNF above has exactly  $d/2$  positive literals, and these clauses encode exactly the maxterms of the corresponding sub-function. Hence each such distinct CNF is a distinct sub-function of  $\text{Int}_{n,d}$  with respect to  $a_1$ . Therefore the number of sub-functions is at least the number of ways to choose  $n$  vectors from  $\mathcal{H}_{d/2}$ .

Since  $d = c \log n$  and  $c > 1$ , for large  $n$ , we have:

$$|\mathcal{H}_{d/2}| = \binom{d}{d/2} \geq \frac{2^d}{2\sqrt{d/2}} = \frac{n^c}{\sqrt{2c \log n}} > n$$

Thus, the number of distinct subfunctions is:

$$s_1 \geq \binom{|\mathcal{H}_{d/2}|}{n} \geq \binom{2^d/2\sqrt{d/2}}{n} \geq \left(\frac{2^d}{n\sqrt{2d}}\right)^n \geq \left(\frac{2^{d-1}}{n\sqrt{d}}\right)^n$$

**Upper bound.** To upper bound the number of distinct sub-functions on  $a_1$ , it suffices to bound the number of different settings to the vectors  $a_2, \dots, a_n, b_1, \dots, b_n$ . This is at most  $2^{(2n-1)d} \leq 2^{2nd}$ . ◀

► **Lemma 27.** *Let  $d \leq \log n$ . The following inequalities hold for the number of distinct sub-functions  $s_i$  of  $\text{Int}_{n,d}$  with respect to any  $a_i$  or  $b_i$ :*

$$2^{\binom{d}{d/2}} \leq s_i \leq 2^{2nd}$$

**Proof.** As described in proof of Lemma 26, we can set the  $b_i$ s to obtain a monotone CNF with clauses on variables of  $a_1$ . Observe that since  $d \leq \log n$ , there are at most  $2^d \leq n$  many possible monotone clauses. Thus, we have sufficiently many  $b_i$  to encode any monotone CNF on the variables of  $a_i$ . Hence, *every* monotone function on the  $d$  variables of  $a_1$  are sub-functions of  $\text{Int}_{n,d}$  with respect to  $a_1$ . The number of monotone functions on  $d$  variables is at least the number of slice functions on the  $d/2$ 'th slice. This is  $2^{\binom{d}{d/2}}$ .

The upper bound is exactly as in proof of Lemma 26.  $\blacktriangleleft$

**Proof of Theorem 25.** Partition the  $2nd$  variables into parts that correspond to each vector. i.e., for all  $i \in [n]$ , the part  $Y_i = a_i$  and for  $j \in \{n+1, \dots, 2n\}$ , the part  $Y_j = b_j$ .

The case of  $d \geq c \log n$  for  $c > 1$  is as follows. Combining Lemma 26 and Theorem 24, we obtain the following.

$$\begin{aligned} L(\text{Int}_{n,d}) &\geq \frac{1}{4} \sum_{i=1}^{2n} \log \left( \frac{2^{d-1}}{n\sqrt{d}} \right)^n = \frac{n^2}{2} (d-1 - (\log n + 1/2 \log d)) \in \Omega(n^2 d) \\ BP(\text{Int}_{n,d}) &\geq \epsilon \sum_{i=1}^{2n} \frac{\log \left( \frac{2^{d-1}}{n\sqrt{d}} \right)^n}{\log \log 2^{2nd}} = \epsilon 2n \frac{n(d-1 - (\log n + 1/2 \log d))}{\log 2nd} \in \Omega \left( \frac{n^2 d}{\log(nd)} \right) \end{aligned}$$

For the case of  $d \leq \log n$ , we use Lemma 27 in Theorem 24:

$$\begin{aligned} L(\text{Int}_{n,d}) &\geq \frac{1}{4} \sum_{i=1}^{2n} \log 2^{\binom{d}{d/2}} = \frac{n}{2} \binom{d}{d/2} \geq \frac{n2^d}{4\sqrt{d/2}} \in \Omega \left( \frac{n2^d}{\sqrt{d}} \right) \\ BP(\text{Int}_{n,d}) &\geq \epsilon \sum_{i=1}^{2n} \frac{\log 2^{\binom{d}{d/2}}}{\log \log 2^{2nd}} \in \Omega \left( \frac{n2^d}{\sqrt{d} \log(2nd)} \right) \end{aligned}$$

$\blacktriangleleft$

► **Remark 28.** We note that Kane and Williams [26] established a lower bound of  $L(\text{Int}_{n,d}) \geq \Omega \left( \min \left( \frac{n^2}{\log d}, \frac{n2^d}{\sqrt{d} \log d} \right) \right)$  for formulas computing  $\text{Int}_{n,d}$  and a lower bound of  $BP(\text{Int}_{n,d}) \geq \Omega \left( \min \left( \frac{n^2}{\log d \log(nd)}, \frac{n2^d}{\sqrt{d} \log d \log(nd)} \right) \right)$  for branching programs computing  $\text{Int}_{n,d}$ . For  $c > 1$  and  $d \geq c \log n$  our result improves this bound for both formulas and branching programs by a factor of  $d \log d$ . When  $d \leq \log n$  the improvement is by a factor of  $\log d$  in both the models.

## 6 Monotone Upper Bounds for $\text{Int}_{n,d}$

In this section, we show a construction of monotone circuits to compute  $\text{Int}$ . We first need the following characterization of 0-inputs of  $\text{Int}_{n,d}$ .

► **Lemma 29.** *Let  $x = (\mathcal{A}, \mathcal{B})$  be an input of  $\text{Int}_{n,d}$ . Then  $\text{Int}_{n,d}(x) = 0$  if and only if there exists an ordered partition  $\mathcal{P} = (P_1, P_2)$  of  $[d]$  into two parts such that there exists  $a \in \mathcal{A}, b \in \mathcal{B}$  satisfying  $a \cap P_1 = \emptyset$  and  $b \cap P_2 = \emptyset$*

**Proof.** “ $\implies$ ”: Let  $x = (\mathcal{A}, \mathcal{B})$  be a 0-input of  $\text{Int}_{n,d}$ . From Definition 3 there must exist  $a \in \mathcal{A}, b \in \mathcal{B}$  such that  $a \cap b = \emptyset$ . Hence for all  $j \in [d]$ , either  $a_j = 0$  or  $b_j = 0$  or both. For every  $j \in [d]$ , we include  $j$  in the set  $P_1$  if  $a_j = 0$ , and in  $P_2$  if  $b_j = 0$ . If both  $a_j$  and  $b_j$  are 0, then we include  $j$  in only  $P_1$ . Therefore each  $j \in [d]$  belongs to exactly one of  $P_1$  and  $P_2$ . This results is an ordered partition of  $[d]$  into two parts namely  $P_1$  and  $P_2$ . It is now straightforward to observe that  $\mathcal{P} = (P_1, P_2)$  satisfies the claim.

“ $\Leftarrow$ ”: Let  $x = (\mathcal{A}, \mathcal{B})$  be an input of  $\text{Int}_{n,d}$ . Suppose  $\mathcal{P} = (P_1, P_2)$  is an ordered partition of  $[d]$  into two parts such that  $a \cap P_1 = \emptyset$  and  $b \cap P_2 = \emptyset$ . Then, consider any  $j \in [d]$ . Since  $\mathcal{P}$  is a partition of  $[d]$ , either  $j \in P_1$  or  $j \in P_2$ . If  $j \in P_1$ , then  $a_j = 0$ , else  $j \in P_1$  and  $b_j = 0$ . Hence  $x$  is a 0-input of  $\text{Int}_{n,d}$ .  $\blacktriangleleft$

► **Theorem 30.** *The function  $\text{Int}_{n,d}$  can be computed by a monotone circuit of size at most  $O(\min\{2^d nd, n^2 d\})$ .*

**Proof.** We will construct two circuits  $C_1$ , and  $C_2$ , computing  $\text{Int}_{n,d}$  with sizes  $O(2^d nd)$ , and  $O(n^2 d)$  respectively.

The circuit  $C_1$  is a brute-force through all possible ordered partitions to check if any of them satisfy Lemma 29. Let  $\mathcal{S}$  be the set of all ordered partitions of  $[d]$  into two parts. The following circuit outputs 0 if for some fixed ordered partition  $\mathcal{P} = (P_1, P_2)$ , there exists  $a \in \mathcal{A}, b \in \mathcal{B}$  such that  $a \cap P_1 = \emptyset$  and  $b \cap P_2 = \emptyset$ .

$$C_{\mathcal{P}} = \left( \bigwedge_{a \in \mathcal{A}} \bigvee_{j \in P_1} a_j \right) \vee \left( \bigwedge_{b \in \mathcal{B}} \bigvee_{j \in P_2} b_j \right)$$

Finally to check if there exists an ordered partition  $\mathcal{P}$  such that  $C_{\mathcal{P}}$  outputs 0 we AND over all ordered partitions of  $[d]$  into two parts.

$$C_1 = \bigwedge_{\mathcal{P} \in \mathcal{S}} C_{\mathcal{P}}$$

Here each  $C_{\mathcal{P}}$  is of size at most  $O(nd)$  and  $|\mathcal{S}| = O(2^d)$ . Therefore size of  $C$  is at most  $O(2^d nd)$ .

The circuit  $C_2$  is a brute-force search to match Definition 3:

$$C_2 = \bigwedge_{a \in \mathcal{A}, b \in \mathcal{B}} \bigvee_{j \in [d]} a_j \wedge b_j$$

Since, each  $|\mathcal{A}| = |\mathcal{B}| = n$ , the size of  $C$  is  $O(n^2 d)$ .  $\blacktriangleleft$

## 7 Jukna's criterion for $\text{Int}_{n,d}$

Jukna [25] (Section 9.4) gives a general criterion to obtain lower bounds for monotone circuits based on the monotone switching lemma of Berg and Ulfberg [11]. This criterion is as follows.

► **Theorem 31.** *Assume that  $f$  is a monotone Boolean function on  $n$  variables and has monotone circuits of size  $t$ . Then, for any  $2 \leq r, s \leq n$ , there is an exact  $s$ -CNF  $\varphi$ , an exact  $r$ -DNF  $\psi$  and a subset  $X'$  of the variables of size at most  $s - 1$  such that*

- $\varphi$  has at most  $t \cdot (r - 1)^s$  clauses and  $\psi$  has at most  $t \cdot (s - 1)^r$  terms,<sup>10</sup> and
- $\varphi \leq f$  or  $f \leq \psi \vee \bigvee_{x \in X'} x$ .

To use the above for a lower bound for  $\text{Int}_{n,d}$ , we need to choose a suitable  $r, s$  so that the conclusion is not true for  $\text{Int}_{n,d}$ . Unfortunately, this is not possible as long as  $c_0 \log n \leq d = n^{o(1)}$  for a large enough absolute constant  $c_0$ . We argue this below.

Our argument rests on the following claims.

<sup>10</sup> An *Exact  $s$ -CNF* is one where every clause has exactly  $s$  distinct variables. Exact  $r$ -DNFs are defined similarly.

▷ **Claim 32.** For any  $2 \leq r \leq n$ , there is an exact  $r$ -DNF  $\psi_0$  with at most  $d^r$  terms such that  $\text{Int}_{n,d} \leq \psi_0$ .

▷ **Claim 33.** For any  $s < d/2$ , there is an exact  $s$ -CNF  $\varphi_1$  with at most  $nd \cdot 2^{O(s)}$  clauses such that  $\varphi_1 \leq \text{Int}_{n,d}$ .

▷ **Claim 34.** For any  $n, d$ ,  $\text{Int}_{n,d}$  can be written as an exact  $d$ -CNF  $\varphi_0$  with at most  $n^2 \cdot 2^{O(d)}$  clauses.

Given the above claims, we can prove that for any choice of  $r, s \in \{2, \dots, 2nd\}$ ,  $\text{Int}_{n,d}$  satisfies the conclusion of Theorem 31 for  $t = n^{1+o(1)}$ , and hence Theorem 31 cannot be used to prove a near-quadratic lower bound. This is done as follows.

- If  $s \geq d + 1$ , we note that  $\text{Int}_{n,d} \leq \bigvee_{i=1}^d a_{1i}$  and hence there is a subset  $X'$  of at most  $s - 1$  variables such that  $\text{Int}_{n,d} \leq \bigvee_{x \in X'} x$ . For the remainder of the argument, assume that  $s \leq d$ .
- If  $r \leq r_0$  for a large enough constant  $r_0$  (chosen below), then we use Claim 32, which gives us an exact  $r$ -DNF  $\psi_0$  such that  $\text{Int}_{n,d} \leq \psi_0$ . Note that  $d^r \leq d^{r_0} \leq n^{o(1)}$  as  $d \leq n^{o(1)}$  and hence we are done. For the remainder of the argument, assume that  $r > r_0$ .
- If  $s \in [d/2, d]$ , then we consider the instance of  $\text{Int}_{n,s}$  obtained by setting the last  $d - s$  entries in each vector in the input lists  $A, B$  to 0. Note that  $\text{Int}_{n,s} \leq \text{Int}_{n,d}$  and by Claim 34, we can write it as an exact  $s$ -CNF  $\varphi$  with at most  $n^2 \cdot 2^{O(s)}$  clauses. As  $s \geq d/2$  and  $d \geq c_0 \log n$ , the number of clauses is at most  $(r - 1)^s$  as long as  $r_0$  and  $c_0$  are large enough. Hence we are done in this case.
- If  $s < d/2$ , then we use Claim 33 to directly obtain an exact  $s$ -CNF  $\varphi_1 \leq \text{Int}_{n,d}$  with at most  $nd \cdot 2^{O(s)} \leq n^{1+o(1)} \cdot (r - 1)^s$  as  $d \leq n^{o(1)}$  and for a large enough constant  $r_0$ . This finishes the argument.

## 7.1 Proof of Claims

In this section, we sketch the proofs of the above claims, all of which are straightforward.

**Proof of Claim 32.** This can be done by writing down an exact  $r$ -DNF  $\psi_0$  that is zero exactly when one of the vectors  $a_1, \dots, a_r$  in the list  $A$  is 0. Note that this implies that  $\text{Int}_{n,d}$  is also 0 and hence  $\text{Int}_{n,d} \leq \psi_0$ . We can write down such a  $\psi_0$  by

$$\psi_0 = \bigvee_{i_1, \dots, i_r \in [d]} a_{1,i_1} \wedge a_{2,i_2} \cdots \wedge a_{r,i_r}.$$

This clearly has  $d^r$  terms. ◀

**Proof of Claim 34.** This can be done by writing out the monotone depth-3 circuit  $C_2$  from the previous section as an exact  $d$ -CNF. More precisely, we have

$$\varphi_0(A, B) = \bigwedge_{i,j \in [n]} \bigwedge_{S \subseteq [d]} \left( \bigvee_{p \in S} a_{i,p} \vee \bigvee_{q \notin S} b_{j,q} \right).$$

This has the required properties. ◀

**Proof of Claim 33.** We show that for any set  $Y = \{y_1, \dots, y_d\}$  of  $d$  Boolean variables, there is an exact  $s$ -CNF  $\varphi'$  with  $d \cdot 2^{O(s)}$  clauses such that

$$\varphi'(y) = 1 \implies |y| > d/2. \quad (7)$$

Given this, we can construct  $\varphi_0$  as follows

$$\varphi_1 = \bigwedge_{i=1}^n \varphi'(a_i) \wedge \bigwedge_{j=1}^n \varphi'(b_j).$$

If  $\varphi_0(A, B) = 1$ , then each vector has Hamming weight greater than  $d/2$ , implying that all pairs intersect and hence  $\varphi_0 \leq \text{Int}_{n,d}$ . Moreover, the total size is  $2nd \cdot 2^{O(s)} = nd \cdot 2^{O(s)}$ .

It remains to construct  $\varphi'$ . This is done by a standard random argument. Pick  $T$  clauses  $C_1, \dots, C_T$  of size exactly  $s$  independently and uniformly at random from among the variables  $Y$  and set  $\varphi' = C_1 \wedge C_2 \wedge \dots \wedge C_T$ . For any setting  $y$  to the variables of weight at most  $d/2$ , the probability that a single clause is not satisfied is at least

$$\frac{\binom{d/2}{s}}{\binom{d}{s}} \geq \frac{(d/2s)^s}{(ed/s)^s} = \frac{1}{2^{O(s)}}$$

where the first inequality uses Stirling approximations. Thus, the probability that  $\varphi'(y) = 1$  is at most

$$\left(1 - \frac{1}{2^{O(s)}}\right)^T < \frac{1}{2^d}$$

for  $T = d \cdot 2^{O(s)}$ . By the probabilistic method, this implies that there is a  $\varphi'$  with the required properties. This finishes the argument.  $\blacktriangleleft$

---

## References

- 1 Amir Abboud, Arturs Backurs, and Virginia Vassilevska Williams. Tight hardness results for LCS and other sequence similarity measures. In *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.
- 2 Amir Abboud, Karl Bringmann, Holger Dell, and Jesper Nederlof. More consequences of falsifying SETH and the orthogonal vectors conjecture. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 253–266. ACM, 2018. doi:10.1145/3188745.3188938.
- 3 Amir Abboud, Karl Bringmann, Danny Hermelin, and Dvir Shabtay. SETH-based lower bounds for subset sum and bicriteria path. *ACM Trans. Algorithms*, 18(1), jan 2022. doi:10.1145/3450524.
- 4 Amir Abboud, Ryan Williams, and Huacheng Yu. More applications of the polynomial method to algorithm design. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '15, pages 218–230, USA, 2015. Society for Industrial and Applied Mathematics.
- 5 Noga Alon and Ravi B. Boppana. The monotone circuit complexity of boolean functions. *Comb.*, 7(1):1–22, 1987. doi:10.1007/BF02579196.
- 6 Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995. doi:10.1145/210332.210337.
- 7 Kazuyuki Amano.  $k$ -subgraph isomorphism on  $\text{ac}^0$  circuits. *Comput. Complex.*, 19(2):183–210, 2010. URL: <https://doi.org/10.1007/s00037-010-0288-y>, doi:10.1007/S00037-010-0288-Y.

- 8 Kazuyuki Amano and Akira Maruoka. A superpolynomial lower bound for a circuit computing the clique function with at most  $(1/6)\log \log n$  negation gates. *SIAM Journal on Computing*, 35(1):201–216, 2005. [arXiv:https://doi.org/10.1137/S0097539701396959](https://doi.org/10.1137/S0097539701396959), [doi:10.1137/S0097539701396959](https://doi.org/10.1137/S0097539701396959).
- 9 Arturs Backurs and Piotr Indyk. Edit distance cannot be computed in strongly subquadratic time (unless SETH is false). In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 51–58. ACM, 2015. [doi:10.1145/2746539.2746612](https://doi.org/10.1145/2746539.2746612).
- 10 Arturs Backurs, Liam Roditty, Gilad Segal, Virginia Vassilevska Williams, and Nicole Wein. Towards tight approximation bounds for graph diameter and eccentricities. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 267–280. ACM, 2018. [doi:10.1145/3188745.3188950](https://doi.org/10.1145/3188745.3188950).
- 11 Christer Berg and Staffan Ulfberg. Symmetric approximation arguments for monotone lower bounds without sunflowers. *Comput. Complex.*, 8(1):1–20, 1999. URL: <https://doi.org/10.1007/s000370050017>, [doi:10.1007/S000370050017](https://doi.org/10.1007/S000370050017).
- 12 Jaroslaw Blasiok and Linus Meierhöfer. Hardness of clique approximation for monotone circuits. In Srikanth Srinivasan, editor, *40th Computational Complexity Conference, CCC 2025, Toronto, Canada, August 5-8, 2025*, LIPIcs, pages 4:1–4:20. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2025. URL: <https://doi.org/10.4230/LIPIcs.CCC.2025.4>, [doi:10.4230/LIPIcs.CCC.2025.4](https://doi.org/10.4230/LIPIcs.CCC.2025.4).
- 13 Chris Calabro, Russell Impagliazzo, and Ramamohan Paturi. On the exact complexity of evaluating quantified  $k$ -cnf. In *Parameterized and Exact Computation - 5th International Symposium, IPEC 2010, Chennai, India, December 13-15, 2010. Proceedings*, volume 6478 of *Lecture Notes in Computer Science*, pages 50–59. Springer, 2010. [doi:10.1007/978-3-642-17493-3\\_7](https://doi.org/10.1007/978-3-642-17493-3_7).
- 14 Bruno Cavalar, Mika Göös, Artur Riazanov, Anastasia Sofronova, and Dmitry Sokolov. Monotone circuit complexity of matching, 2025. URL: <https://arxiv.org/abs/2507.16105>, [arXiv:2507.16105](https://arxiv.org/abs/2507.16105).
- 15 Bruno Pasqualotto Cavalar, Théo Borém Fabris, Partha Mukhopadhyay, Srikanth Srinivasan, and Amir Yehudayoff. Negations are powerful even in small depth. *Electron. Colloquium Comput. Complex.*, TR25, 2025. URL: <https://eccc.weizmann.ac.il/report/2025/219>, [arXiv:TR25-219](https://arxiv.org/abs/2507.16105).
- 16 Bruno Pasqualotto Cavalar, Mrinal Kumar, and Benjamin Rossman. Monotone circuit lower bounds from robust sunflowers. *Algorithmica*, 84(12):3655–3685, 2022. URL: <https://doi.org/10.1007/s00453-022-01000-3>, [doi:10.1007/S00453-022-01000-3](https://doi.org/10.1007/S00453-022-01000-3).
- 17 Timothy M. Chan and R. Ryan Williams. Deterministic apsp, orthogonal vectors, and more: Quickly derandomizing razborov-smolensky. *ACM Trans. Algorithms*, 17(1), December 2021. [doi:10.1145/3402926](https://doi.org/10.1145/3402926).
- 18 Tameem Choudhury and KartEEK Sreenivasaiiah. Depth-3 circuit lower bounds for  $k$ -ov. In Olaf Beyersdorff, Mamadou Moustapha Kanté, Orna Kupferman, and Daniel Lokshtanov, editors, *41st International Symposium on Theoretical Aspects of Computer Science, STACS 2024, Clermont-Ferrand, France, March 12-14, 2024*, volume 289 of *LIPIcs*, pages 25:1–25:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024. URL: <https://doi.org/10.4230/LIPIcs.STACS.2024.25>, [doi:10.4230/LIPIcs.STACS.2024.25](https://doi.org/10.4230/LIPIcs.STACS.2024.25).
- 19 Tameem Choudhury and KartEEK Sreenivasaiiah. Depth-3 circuit lower bounds for  $k$ -ov. *ACM Trans. Comput. Theory*, May 2026. Just Accepted. [doi:10.1145/3812802](https://doi.org/10.1145/3812802).
- 20 Susanna F. de Rezende and Marc Vinyals. Lifting with colourful sunflowers. In Srikanth Srinivasan, editor, *40th Computational Complexity Conference, CCC 2025, Toronto, Canada, August 5-8, 2025*, LIPIcs, pages 36:1–36:19. Schloss Dagstuhl - Leibniz-Zentrum

- für Informatik, 2025. URL: <https://doi.org/10.4230/LIPIcs.CCC.2025.36>, doi:10.4230/LIPIcs.CCC.2025.36.
- 21 Paul Erdős and Richard Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 1(1):85–90, 1960.
  - 22 Jiawei Gao, Russell Impagliazzo, Antonina Kolokolova, and Ryan Williams. Completeness for first-order properties on sparse structures with algorithmic applications. *ACM Trans. Algorithms*, 15(2):23:1–23:35, 2019. doi:10.1145/3196275.
  - 23 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.
  - 24 Stasys Jukna. Combinatorics of monotone computations. *Comb.*, 19(1):65–85, 1999. URL: <https://doi.org/10.1007/s004930050046>, doi:10.1007/S004930050046.
  - 25 Stasys Jukna. *Boolean Function Complexity - Advances and Frontiers*, volume 27 of *Algorithms and combinatorics*. Springer, 2012. doi:10.1007/978-3-642-24508-4.
  - 26 Daniel M. Kane and Richard Ryan Williams. The orthogonal vectors conjecture for branching programs and formulas. In *10th Innovations in Theoretical Computer Science Conference, ITCS 2019, January 10-12, 2019, San Diego, California, USA*, volume 124 of *LIPIcs*, pages 48:1–48:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019. doi:10.4230/LIPIcs.ITCS.2019.48.
  - 27 Gyula Katona. A theorem of finite sets. In *Classic Papers in Combinatorics*, pages 381–401. Springer, 1968.
  - 28 Joseph B. Kruskal. 12. *The Number of Simplices in a Complex*, pages 251–278. University of California Press, Berkeley, 2024. URL: <https://doi.org/10.1525/9780520319875-014> [cited 2026-02-03], doi:doi:10.1525/9780520319875-014.
  - 29 Eduard Ivanovich Nechiporuk. On a boolean function. In *Doklady Akademii Nauk*, volume 169, pages 765–766. Russian Academy of Sciences, 1966.
  - 30 Anup Rao. The story of sunflowers. *Journal of the London Mathematical Society*, 113(1):e70380, 2026. URL: <https://londmathsoc.onlinelibrary.wiley.com/doi/abs/10.1112/jlms.70380>, doi:10.1112/jlms.70380.
  - 31 Alexander Razborov. Lower bounds on the monotone complexity of some boolean function. In *Soviet Math. Dokl.*, volume 31, pages 354–357, 1985.
  - 32 Benjamin Rossman. The monotone complexity of  $k$ -clique on random graphs. *SIAM Journal on Computing*, 43(1):256–279, 2014. arXiv:<https://doi.org/10.1137/110839059>, doi:10.1137/110839059.
  - 33 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.
  - 34 Ryan Williams. The orthogonal vectors conjecture and non-uniform circuit lower bounds. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 1372–1387. IEEE, 2024. doi:10.1109/FOCS61266.2024.00088.
  - 35 Virginia Vassilevska Williams. On some fine-grained questions in algorithms and complexity. In *Proceedings of the International Congress of Mathematicians: Rio de Janeiro 2018*, pages 3447–3487. World Scientific, 2018.