

# Unambiguous DNFs from Hex

Shalev Ben-David  
University of Waterloo

Mika Göös  
EPFL

Siddhartha Jain  
EPFL

Robin Kothari  
Microsoft Quantum

February 16, 2021

**Abstract.** We exhibit an unambiguous  $k$ -DNF formula that requires CNF width  $\tilde{\Omega}(k^{1.5})$ . Our construction is inspired by the board game Hex and it is vastly simpler than previous ones, which achieved at best an exponent of 1.22. Our result is known to imply several other improved separations in query and communication complexity (e.g., clique vs. independent set problem) and graph theory (Alon–Saks–Seymour problem).

## 1 Three puzzles

**First formulation.** An  $n$ -variate DNF formula  $F = C_1 \vee \dots \vee C_m$  is said to be *unambiguous* if for every input  $x \in \{0, 1\}^n$  at most one of the conjunctions  $C_i$  evaluates to true,  $C_i(x) = 1$ . If we think of the DNF formula as expressing its set of 1-inputs  $F^{-1}(1)$  as a union of subcubes  $C_i^{-1}(1)$ , then  $F$  is unambiguous precisely when the subcubes are pairwise disjoint. Unambiguity is a severe structural restriction on DNFs. In particular, every unambiguous DNF formula of bounded *width* (defined as the maximum number of literals in a conjunction) can be written equivalently as a bounded-width CNF formula. Namely, we have the following folklore fact [Göös15, §III].

**Fact 1.** *Every unambiguous  $k$ -DNF can be written equivalently as a  $k^2$ -CNF.*

In this paper, we ask: Can this quadratic relationship be improved? Are there  $k$ -DNFs that require CNFs of width much larger than  $k$ , perhaps even  $\Omega(k^2)$ ? More formally, for a boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$  we define the following standard complexity measures.

- 1-certificate complexity  $C_1(f)$  is the least  $k$  such that  $f$  can be written as a  $k$ -DNF;
- 0-certificate complexity  $C_0(f)$  is the least  $k$  such that  $f$  can be written as a  $k$ -CNF;
- unambiguous 1-certificate complexity  $UC_1(f)$  is the least  $k$  such that  $f$  can be written as an unambiguous  $k$ -DNF.

**Puzzle I.** *For  $\alpha > 1$ , does there exist a boolean function  $f$  with  $C_0(f) \geq UC_1(f)^{\alpha-o(1)}$ ?*

Here we abused terminology: instead of a single boolean function we really mean an infinite sequence of functions  $f_n$  satisfying  $C_0(f_n) = \omega(1)$  as  $n \rightarrow \infty$ . **Puzzle I** was first asked in [Göös15], although an analogous question had been studied in communication complexity (under the name *clique vs. independent set*; see Section 2.1) since Yannakakis [Yan91]. The paper [Göös15] gave a complicated recursive construction achieving an exponent  $\alpha \approx 1.12$ . This was subsequently optimised (but not simplified) in [BHT17] improving the exponent to  $\alpha \approx 1.22$ . Our main result, stated as **Theorem 1** in Section 2, gives a new simple construction with  $\alpha = 1.5$ .

**Second formulation.** In order to separate boolean function complexity measures it is often a good idea to proceed in two steps: First construct a partial boolean function  $f: \{0, 1\}^n \rightarrow \{0, 1, *\}$  where some inputs  $x$  are *undefined*,  $f(x) = *$ . Then modify  $f$  into a *total* function by eliminating all the  $*$ -inputs. We proceed to formulate an appropriate partial function version of [Puzzle I](#).

We recall the notion of a *certificate*, adapted here for a partial function  $f: \{0, 1\}^n \rightarrow \{0, 1, *\}$ . Let  $\Sigma \subseteq \{0, 1, *\}$  be a subset of output symbols. We write for short  $0, 1, \bar{0}, \bar{1}$  for the output sets  $\{0\}, \{1\}, \{1, *\}, \{0, *\}$ . A partial input  $\rho \in \{0, 1, *\}^n$  is a  $\Sigma$ -*certificate* for  $x \in \{0, 1\}^n$  if  $\rho$  is consistent with  $x$  and for every input  $x'$  consistent with  $\rho$  we have  $f(x') \in \Sigma$ . The *size* of  $\rho$ , denoted  $|\rho|$ , is the number of its non- $*$  entries. The  $\Sigma$ -*certificate complexity* of  $x$ , denoted  $C_\Sigma(f, x)$ , is the least size of a  $\Sigma$ -certificate for  $x$ . The  $\Sigma$ -*certificate complexity* of  $f$ , denoted  $C_\Sigma(f)$ , is the maximum of  $C_\Sigma(f, x)$  over all  $x \in f^{-1}(\Sigma)$ ; this definition is consistent with the one given at the start of this section. Finally, we define *certificate complexity*  $C(f)$  as  $\max\{C_0(f), C_1(f)\}$ .

**Puzzle II.** *For  $\alpha > 1$ , does there exist a partial function  $f$  together with an  $x \in f^{-1}(*)$  such that both  $C_{\bar{0}}(f, x)$  and  $C_{\bar{1}}(f, x)$  are at least  $C(f)^{\alpha-o(1)}$ ?*

We will show in [Theorem 2](#) that [Puzzle I](#) and [Puzzle II](#) are in fact equivalent: solving one with an exponent  $\alpha$  will imply a solution to the other one with the same  $\alpha$ . The implication [I](#) $\Rightarrow$ [II](#) is easy while the converse (converting a partial function into a total one) is non-trivial and uses the *cheat sheet* framework introduced in [\[ABK16\]](#). Consequently, we feel that [II](#) is the most fruitful formulation to attack and that is indeed how our new separation result is obtained.

**Third formulation.** We present one more equivalent formulation using purely graph theoretic language. While this version is not needed for our separation result, we include it for aesthetic reasons. Let  $G = (V, E)$  be a hypergraph. We say  $G$  is *intersecting* if every two edges  $e, e' \in E$  intersect,  $e \cap e' \neq \emptyset$ . A subset  $U \subseteq V$  is a *hitting set* for  $G$  if  $U$  intersects every edge  $e \in E$ . Moreover,  $U$  is *c-monochromatic* for a colouring  $c: V \rightarrow \{0, 1\}$  if  $c$  is constant on  $U$ . Finally, we define the *rank* of  $G$ , denoted  $r(G)$ , as the maximum size  $|e|$  of an edge  $e \in E$ .

**Puzzle III.** *For  $\alpha > 1$ , does there exist an intersecting hypergraph  $G = (V, E)$  together with a colouring  $c: V \rightarrow \{0, 1\}$  such every  $c$ -monochromatic hitting set has size at least  $r(G)^{\alpha-o(1)}$ ?*

[Puzzle III](#) obscures the complexity-theoretic origins of the problem, thereby rendering it increasingly seductive for, say, an unsuspecting audience of combinatorialists (cf. [\[Raz11\]](#)). In fact, we found all three formulations and proved them equivalent already in late 2015, and since then we have been deploying the camouflaged variant [III](#) on several occasions, including, notably and most unsuccessfully, at an open problem seminar at the Institute for Advanced Study in 2018.

## 2 Our contributions

Our main results are as follows; here, the notation  $\tilde{\Omega}(n)$  hides  $\text{poly}(\log n)$  factors.

**Theorem 1.** *There exists a boolean function  $f$  with  $C_0(f) \geq \tilde{\Omega}(\text{UC}_1(f)^{1.5})$ .*

**Theorem 2.** *Puzzles [I](#), [II](#), [III](#) are equivalent: if one of them can be solved with exponent  $\alpha$ , then all of them can be solved with exponent  $\alpha$  (up to factors logarithmic in input length).*

To prove [Theorem 1](#) we describe a simple boolean function inspired by the board game Hex. Our construction, in [Section 3](#), is phrased as a solution to [Puzzle II](#). Hence [Theorem 1](#) follows from the equivalences in [Theorem 2](#), which are in turn proved in [Section 4](#). We next discuss how our results imply several other improved separations in query and communication complexity.

## 2.1 Applications: Clique vs. independent set and Alon–Saks–Seymour

The original motivation for studying [Puzzle I](#) in [\[Gö15\]](#) was that its solutions imply lower bounds for two well-studied problems.

- *Clique vs. independent set problem* [\[Yan91\]](#). This two-party communication problem is defined relative to an  $n$ -vertex graph  $G = (V, E)$ : Alice gets a clique  $x \subseteq V$ , Bob gets an independent set  $y \subseteq V$ , and their goal is to output  $\text{CIS}_G(x, y) := |x \cap y| \in \{0, 1\}$ .
- *Alon–Saks–Seymour problem*. For a graph  $G$ , how large can the chromatic number  $\chi(G)$  be compared to the biclique partition number  $\text{bp}(G)$  (minimum number of complete bipartite graphs needed to partition the edges of  $G$ )?

A surprising connection here is that proving lower bounds on the conondeterministic communication complexity of clique vs. independent set is equivalent to constructing separations for the Alon–Saks–Seymour problem; see Bousquet et al. [\[BLT14\]](#) for an excellent survey of this connection. Huang and Sudakov [\[HS12\]](#) were the first to find a polynomial separation between  $\chi(G)$  and  $\text{bp}(G)$ , which disproved a conjectured linear relationship due to Alon, Saks, and Seymour. Subsequent work has exhibited larger polynomial separations [\[Ama14, SA15\]](#) and even quasi-polynomial ones [\[Gö15, BHT17\]](#). If we plug [Theorem 1](#) into the lifting framework of [\[Gö15, GLM<sup>+</sup>16\]](#) we get the following improvements over the previous-best lower bounds with exponent 1.22 [\[BHT17\]](#).

**Corollary 3.** *There exist a graph  $G$  such that  $\text{CIS}_G$  requires  $\tilde{\Omega}(\log^{1.5} n)$  bits of conondeterministic communication. Equivalently, there exist a graph  $H$  such that  $\chi(H) \geq \exp(\tilde{\Omega}(\log^{1.5} \text{bp}(H)))$ .*

Let us pause here to appreciate how long a chain of reductions we have created to solve a graph theoretic problem by a reduction to another (hyper)graph theoretic problem, but fundamentally passing through complexity theory. That is, we have

1. Alon–Saks–Seymour problem, reduces to
2. clique vs. independent set problem, reduces to
3. [Puzzle I](#): separation  $C_0 \gg UC_1$  in query complexity, reduces to
4. [Puzzle II](#): separation  $C_{\bar{0}}, C_{\bar{1}} \gg C$  for a partial function, reduces to
5. [Puzzle III](#): 2-colouring an intersecting hypergraph.

Reduction 2-to-3 uses a lifting theorem (which is not known to have a converse) and 3-to-4 uses cheat sheets—both of these are inherently query/communication tools that do not have natural counterparts in classical combinatorics. The end result can be phrased as its own graph problem: Given an intersecting hypergraph and a 2-colouring whose monochromatic hitting sets are power- $\alpha$  larger than the rank, construct a graph which is an edge-disjoint union of  $k$  bicliques but which has chromatic number  $\exp(\log^{\alpha-o(1)} k)$ . It sounds to us magical that this is possible!

**Other related work.** It is widely conjectured (e.g., [\[HS12\]](#)) that the exponent 1.5 in [Corollary 3](#) can be improved to 2, which would be optimal [\[Yan91\]](#). Such optimal bounds (up to low-order terms) exist for  $\text{CIS}_G$  in the deterministic [\[GPW18\]](#) and even randomised [\[GJPW18\]](#) communication models. However, these results do not imply any bounds for the conondeterministic complexity and hence neither for the Alon–Saks–Seymour problem.

Given that superpolynomial separations exist for the Alon–Saks–Seymour problem in general, a recent line of work has aimed to find special graph classes where the separation is at most polynomial [\[BLT14, LT16, BLMP18, CS21\]](#). In particular, it remains open whether the separation is polynomial for the class of perfect graphs.

## 2.2 Applications: Separations in query complexity

In query complexity, we get two improved separations involving the well-studied complexity measures *sensitivity*  $s(f)$  and *approximate degree*  $\widetilde{\deg}(f)$  (defined in [Section 5](#)). The first of these corollaries follows automatically from [[BHT17](#), Theorem 1] and the second one we prove in [Section 5](#).

**Corollary 4.** *There exists a boolean function  $f$  with  $C(f) \geq \tilde{\Omega}(s(f)^{2.5})$ .*

**Corollary 5.** *There exists a boolean function  $f$  with  $C(f) \geq \tilde{\Omega}(\widetilde{\deg}(f)^3)$ .*

Previous best separations had exponents 2.22 and 2, respectively. See Aaronson et al. [[ABK<sup>+</sup>21](#)] for an up-to-date survey of the known relationships.

## 2.3 Open problems

We conjecture that the puzzles are soluble with exponent  $\alpha = 2$  matching the upper bound in [Fact 1](#). This would yield near-optimal lower bounds for [Corollary 3](#) and [5](#). There is also an additional motivation to seek a solution with  $\alpha = 2$  (which is not afforded by our solution  $\alpha = 1.5$ ): it would separate  $C(f)$  and  $\deg(f)$  (degree of  $f$  as a multivariate polynomial) quadratically, improving on the power-1.63 separation due to Nisan, Kushilevitz, and Wigderson [[NW95](#)].

## 3 Hex function

In this section, we describe our Hex-inspired solution to [Puzzle II](#) that achieves exponent  $\alpha = 1.5$ . We define a partial boolean function  $\text{HEX}_n: \{0, 1\}^{n \times n} \rightarrow \{0, 1, *\}$  whose  $n^2$ -bit inputs are interpreted as  $n \times n$  boolean matrices. We say that two matrix entries in  $[n] \times [n]$  are *connected* if they are adjacent either horizontally or vertically (but not diagonally). A *1-path* in an input  $x$  is top-to-bottom path of 1-entries, that is, the path starts on a 1-entry in the topmost row, moves along connected 1-entries, and ends on the bottommost row. Similarly, a *0-path* in  $x$  is a left-to-right path of 0-entries. Note that no  $x$  can contain both a 1-path and a 0-path. We define

$$\text{HEX}_n(x) := \begin{cases} 1 & \text{if } x \text{ contains a 1-path of length at most } 2n, \\ 0 & \text{if } x \text{ contains a 0-path of length at most } 2n, \\ * & \text{otherwise.} \end{cases}$$

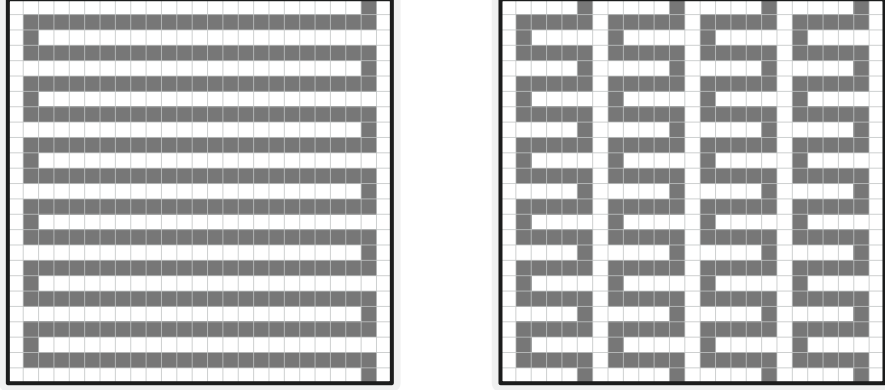
Clearly  $C(\text{HEX}_n) = 2n$ . It remains to prove the following lemma. For simplicity, we drop  $\text{HEX}_n$  from notation and write  $C_\Sigma(x) := C_\Sigma(\text{HEX}_n, x)$ .

**Lemma 6.** *There is an  $x \in \text{HEX}_n^{-1}(*)$  such that both  $C_{\bar{0}}(x)$  and  $C_{\bar{1}}(x)$  are  $\Omega(n^{1.5})$ .*

As a warm-up, we note that it is easy to find inputs  $x$  where one of  $C_{\bar{0}}(x)$  or  $C_{\bar{1}}(x)$  is large, but not both. For example, consider the input  $y$  depicted in [Figure 1a](#) that contains a single spiraling 1-path, call it a *1-spiral* for short. The 1-spiral has length  $\Theta(n^2) > 2n$  and hence  $\text{HEX}_n(y) = *$ .

**Claim 7.**  $C_{\bar{0}}(y) \geq \Omega(n^2)$  and  $C_{\bar{1}}(y) \leq O(n)$ .

*Proof.* For the first claim, we employ a sensitivity argument. Consider any entry  $e \in [n] \times [n]$  in the 1-spiral that is not a *corner* (where the spiral makes a right-angle turn). Denote by  $y^e$  the input  $y$  but with the entry  $e$  flipped (from 1 to 0). Note that flipping  $e$  introduces a short ( $\leq 2n$ ) 0-path in  $y^e$  and thus  $\text{HEX}_n(y^e) = 0$ . It follows that any  $\bar{0}$ -certificate for  $y$  needs to read all the non-corner entries of which there are  $\Theta(n^2)$  many. For the second claim, we note that it suffices to include the five topmost rows in a certificate to prove that any 1-path in  $y$  must be of length  $> 2n$ .  $\square$



(a) input  $y$

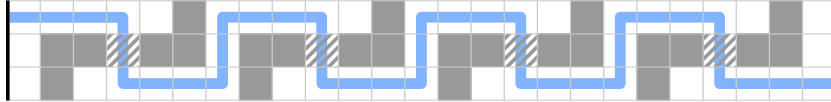
(b) input  $z$

**Figure 1:** Inputs to  $\text{HEX}_n$  are  $n \times n$  boolean matrices. Illustrated are two  $*$ -inputs:  $y$  consists of a single 1-spiral of length  $\Theta(n^2)$ , and  $z$  consists of  $\sqrt{n}$  many 1-spirals of length  $\Theta(n^{1.5})$  each.

We can similarly find an input  $y^*$  with large  $C_{\bar{1}}(y^*)$  and small  $C_{\bar{0}}(y^*)$ . The challenge in [Lemma 6](#) is to find a single input where both  $\bar{0}$ - and  $\bar{1}$ -complexities are large. Our solution is to “balance”  $y$ . Namely, we let  $z$  be the input that consists of  $\sqrt{n}$  many disjoint 1-spirals, each of length  $\Theta(n^{1.5})$ ; see [Figure 1b](#) for an illustration. The following two claims complete the proof of [Lemma 6](#).

**Claim 8.**  $C_{\bar{0}}(z) \geq \Omega(n^{1.5})$ .

*Proof.* We employ a block sensitivity argument. Let  $\ell = \Theta(n^{1.5})$  denote the number of non-corner entries in each 1-spiral of  $z$ . For each  $i \in [\ell]$ , we define a *block*  $B_i \subseteq [n] \times [n]$ ,  $|B_i| = \sqrt{n}$ , as the set that contains the  $i$ -th non-corner entry from each 1-spiral (the non-corners of a spiral are ordered top-to-bottom, say). Denote by  $z^{B_i}$  the input obtained from  $z$  by flipping all the entries in  $B_i$  (from 1 to 0). Flipping any block  $B_i$  introduces a short ( $\leq 2n$ ) 0-path in  $z^{B_i}$  and hence  $\text{HEX}_n(z^{B_i}) = 0$ . For example, in the following illustration, the short 0-path (drawn in blue) is created when we flip the block consisting of the striped entries:



It follows that any  $\bar{0}$ -certificate for  $z$  needs to read at least one entry from each of the blocks. But since the blocks are disjoint and there are  $\ell = \Theta(n^{1.5})$  many of them, the claim is proved.  $\square$

**Claim 9.**  $C_{\bar{1}}(z) \geq \Omega(n^{1.5})$ .

*Proof.* Let  $\rho$  be a partial input consistent with  $z$  that has size  $o(n^{1.5})$ . We show that  $\rho$  cannot be a  $\bar{1}$ -certificate. By averaging, there is some “neglected” 1-spiral such that  $\rho$  reads  $o(n)$  many 0-entries adjacent to the spiral. We will greedily construct a 1-path consistent with  $\rho$  by starting at the top of the neglected spiral and trying to fit a 1-path straight down the matrix. The 0-entries read by  $\rho$  can prevent a direct downward path from working, but every time we encounter such a 0-entry we can avoid it by taking one step to the left or right (following the direction of the spiral). These left/right steps make us waste at most  $o(n)$  extra steps in addition to the  $n$  downward steps. This shows there exists a 1-path of length  $n + o(n) \leq 2n$  consistent with  $\rho$ , and hence  $\rho$  is not a  $\bar{1}$ -certificate.  $\square$

*Remark 10.* It is easy to see that  $C_{\bar{0}}(z)$  and  $C_{\bar{1}}(z)$  are also  $O(n^{1.5})$ . We suspect that  $z$  is in fact extremal for  $\text{HEX}_n$  meaning that no other  $*$ -input can witness an exponent larger than  $\alpha = 1.5$ . However, we have not been able to prove this.

## 4 Equivalences of puzzles

We now prove our three puzzles equivalent (Theorem 2). The proof comprises of four implications, each proved in its own subsection:  $\text{II} \Rightarrow \text{I} \Rightarrow \text{II} \Rightarrow \text{III} \Rightarrow \text{II}$ . This is more implications than strictly necessary, but not all directions are equally good in terms of overheads caused by log factors.

### 4.1 Construction $\text{II} \Rightarrow \text{I}$

**Given:** A partial function  $f: \{0, 1\}^n \rightarrow \{0, 1, *\}$  and an input  $x \in f^{-1}(*)$ .

**Construct:** A total function  $g: \{0, 1\}^{3n^2 \log^2 n} \rightarrow \{0, 1\}$  such that  $C_0(g) \geq \min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\}$  and  $UC_1(g) \leq 3C(f) \log^2 n$ .

**Overview.** The basic idea is that we would like to turn regular, ambiguous certificates for  $f$  into unambiguous collections of certificates for a modified function  $g$ . One way to do so is to give each certificate for  $f$  a unique identification number; then we can require the new inputs to  $g$  to consist of both an input  $z$  to  $f$  and an identification number (written in binary) for a certificate in  $z$ . We will let such a new input  $(z, k)$  evaluate to 1 if the certificate specified by the number  $k$  really is in  $z$ , and we will define  $(z, k)$  to be a 0-input otherwise. Then by reading all of  $k$  and the corresponding certificate in  $z$ , we get an unambiguous certificate for  $(z, k)$  whenever  $(z, k)$  is a 1-input.

This strategy makes 1-certificates unambiguous, but it does not necessarily ensure that the function is hard to certify on 0-inputs. The reason is that for the new function, it is conceivable that we could certify  $(z, k)$  is a 0-input just by reading a few bits of  $k$  and a few bits of  $z$ , but that those few bits suffice to prove that the certificate specified by  $k$  cannot possibly be found in  $z$ . Indeed, it might even be easy to certify that  $(z, k)$  is a 0-input when  $z = x$ , the hard \*-input to  $f$ .

We wish to eliminate this 0-certification strategy so that the new function is hard to certify on at least one 0-input. To do so, we will use the *cheat sheet* framework [ABK16]. The idea is to hide the identification number  $k$  of the certificate in one cell of an array consisting of, say,  $n$  different cells. We choose  $n$  cells because this is large enough so that even reading a single bit from each cell is too expensive. But now that we have hidden  $k$  in one of  $n$  cells, there needs to be a way to find it. So to specify which cell of the array is the “correct” one, the one where we’ve stored  $k$ , we will change the problem to have  $\log n$  different instances of  $f$ , and we will interpret the  $f$ -outputs of these instances as specifying a binary string of length  $\log n$ , which can index a specific cell of our array. Now that there are  $\log n$  copies of  $f$ , the correct array cell will be required to contain identification numbers for  $\log n$  different certificates, one for each instance of  $f$ . Now we define this new function  $g$  to evaluate to 1 if all the  $\log n$   $f$ -inputs are 0- or 1-inputs and if the array cell indexed really contains valid identification numbers of certificates present in the  $f$ -inputs. Otherwise, if this doesn’t hold, we define the input to be a 0-input to  $g$ .

With this construction, the contents of the cell pointed to by the  $\log n$ -bit string of outputs of  $f$ , along with the certificates in that cell form small unambiguous certificates for 1-inputs to  $g$ . On the other hand, the  $g$ -input consisting of  $\log n$  copies of  $x \in f^{-1}(*)$  together with any array content will be a 0-input that is hard to certify: Any certificate must either prove that at least one copy of  $x$  is not a 0-input or not a 1-input, which is expensive to do because we assumed that  $C_{\bar{0}}(f, x)$  and  $C_{\bar{1}}(f, x)$  are large, or else it must prove that none of the cells in the array contain valid certificates, which requires it to read at least one bit from each of the  $n$  cells. We now prove this more formally.

**Formal proof.** A certificate of size  $C(f)$  specifies the indices of up to  $C(f)$  input bits and an assignment to those bits. Since an index can be encoded using  $\log n$  bits, the total number of bits needed to represent a certificate is at most  $\ell := 2C(f)\log n \leq 2n\log n$ . We choose  $k := \log n$  as the number of copies of  $f$  that we will use. Let us define  $g: \{0, 1\}^{kn+2^k k\ell} \rightarrow \{0, 1\}$  on  $kn + 2^k k\ell \leq 3n^2 \log^2 n$  bits. For an input  $z$  to  $g$ , we define  $s_z$  to be the string in  $\{0, 1, *\}^k$  that we get by applying  $f$  to the first  $kn$  bits of  $z$ , interpreted as  $k$  inputs to  $f$ . If  $s_z \notin \{0, 1\}^k$ , we define  $g(z) := 0$ . Otherwise, if  $s_z \in \{0, 1\}^k$ , we interpret the last  $2^k k\ell$  bits of  $z$  as an array of  $2^k$  cells of size  $k\ell$  each, and we let  $C_z \in \{0, 1\}^{k\ell}$  be the contents of the cell indexed by  $s_z$ . We interpret  $C_z$  as specifying  $k$  different certificates for  $f$ , each specified using  $\ell$  bits. We then set  $g(z) := 1$  if each of the  $k$  inputs for  $g$  in the first part of the string  $z$  contains the corresponding certificates specified by  $C_z$  in order. Otherwise, we set  $g(z) := 0$ .

The following two claims verify that this construction has the desired properties.

**Claim 11.**  $C_0(g) \geq \min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\}$ .

*Proof.* Consider the input to  $g$  consisting of  $k$  copies of  $x$ , followed by an all-0 array. Consider any certificate  $c$  for this input. If  $c$  reads fewer than  $\min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\}$  bits, then  $c$  does not certify that  $x$  is not a 0-input or that  $x$  is not a 1-input for any of the  $k$  copies of  $x$ . Moreover,  $c$  also cannot read a bit of each of the  $2^k = n$  array cells, since  $n$  is larger than  $\min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\}$ . Hence there is some array cell, indexed by some string  $s \in \{0, 1\}^k$ , such that  $c$  reads no bits of that array cell. Since  $c$  fails to prove anything about the  $f$ -outputs of the copies of  $x$ , we can find an input  $y$  to  $g$  which is consistent with  $c$  such that the  $f$ -inputs in  $y$  evaluate to  $s$ ; moreover, we can then set the array cell indexed by  $s$  to provide valid certificates for the  $k$  inputs to  $f$  in  $y$ . This causes  $y$  to be a 1-input consistent with  $c$ , contradicting the assumption that  $c$  was a 0-certificate.  $\square$

**Claim 12.**  $UC_1(g) \leq 3C(f)\log^2 n$ .

*Proof.* Intuitively, the contents of the cell referred to by the string  $s_z$  and all the certificates in it together form an unambiguous certificate for  $f$ . So an unambiguous 1-certificate for  $g$  has the following form: first, it reads exactly one certificate for each of the  $k$  inputs to  $f$ ; second, in the array cell indexed by  $s_z$ , the  $\log n$ -bit string of  $f$ -outputs, the certificate reads the entire array cell, and the cell has the property that it contains exactly the same certificates read in the  $k$  inputs to  $f$  (in order). The size of this certificate is  $k(\ell + C(f)) \leq \log n(2C(f)\log n + C(f)) \leq 3C(f)\log^2 n$  where  $kC(f)$  bits are used to specify the certificates for  $k$  copies of  $f$  and  $k\ell$  bits are used to read the full contents of one cell of the array.

It remains to show that the above collection of 1-certificates is unambiguous. We claim that no input  $z$  to  $g$  can have two different certificates of the type we have just described. To see this, suppose otherwise, and let  $\rho_1$  and  $\rho_2$  be two such certificates consistent with  $z$ . Suppose that  $\rho_1$  reads bits in the array cell  $C_1$  and that  $\rho_2$  reads bits in the array cell  $C_2$ . Then since  $\rho_1$  proves that the  $f$ -inputs in  $z$  evaluate to the index of  $C_1$ , and since  $\rho_2$  proves that the  $f$ -inputs in  $z$  evaluate to the index of  $C_2$ , we must have  $C_1 = C_2$ . Since  $\rho_1$  reads all of  $C_1$  and  $\rho_2$  reads all of  $C_2$ , we know that  $\rho_1$  and  $\rho_2$  are identical on the array part of the input. However, this array cell then specifies exactly which bits a certificate in this collection must read from the  $k$  inputs to  $f$ ; it follows that  $\rho_1$  and  $\rho_2$  must be identical.  $\square$



## 4.2 Construction I $\Rightarrow$ II

**Given:** A total function  $f: \{0, 1\}^n \rightarrow \{0, 1\}$ .

**Construct:** A partial function  $g: \{0, 1\}^{2n} \rightarrow \{0, 1, *\}$  and an input  $z \in g^{-1}(*)$  such that  $\min\{C_{\bar{0}}(g, z), C_{\bar{1}}(g, z)\} \geq C_0(f)$  and  $C(g) \leq 2UC_1(f)$ .

Let  $U \subseteq \{0, 1, *\}^n$  be an unambiguous collection of 1-certificates for  $f$  so that

- for every  $x \in f^{-1}(1)$  there is a unique  $\rho_x \in U$  such that  $x$  is consistent with  $\rho_x$ ;
- each  $\rho \in U$  has size  $|\rho| \leq UC_1(f)$ .

The function  $g$  will be defined on inputs  $(x, y) \in \{0, 1\}^{2n}$  where  $x, y \in \{0, 1\}^n$ . If  $x$  is such that  $f(x) = 0$ , we define  $g(x, y) := *$ . Otherwise if  $f(x) = 1$ , we consider the unique  $\rho_x \in U$  consistent with  $x$ : Denote by  $r(\rho_x) \subseteq [n]$  the set of indices  $i \in [n]$  that are read by  $\rho_x$ . We define  $g(x, y) := \bigoplus_{i \in r(\rho_x)} y_i$ , that is, the parity of the bits of  $y$  that are indexed by  $r(\rho_x)$ .

To certify that  $g(x, y) = b$  for  $b \in \{0, 1\}$ , it suffices to read  $\rho_x \in U$  together with the corresponding set of bits  $r(\rho_x)$  in  $y$ . This shows that  $C(g) \leq 2UC_1(f)$ . We then define the hard  $*$ -input by  $z := (x, 0^n)$  where  $x \in f^{-1}(0)$  is any input such that  $C_0(f, x) = C_0(f)$ .

**Claim 13.**  $\min\{C_{\bar{0}}(g, z), C_{\bar{1}}(g, z)\} \geq C_0(f)$ .

*Proof.* Let  $\rho \in \{0, 1, *\}^{2n}$  be a partial input consistent with  $z$  that has size  $|\rho| < C_0(f)$ . Our goal is to show that  $\rho$  is not a  $\bar{1}$ -certificate (showing that  $\rho$  is not a  $\bar{0}$ -certificate is analogous). It is possible that  $\rho$  reads some bits in the first half of the input  $z = (x, 0^n)$  and some bits in the second half. We define a set  $B := \{i \in [n] : i \in r(\rho) \text{ or } i + n \in r(\rho)\}$  that “shifts” all the query positions in the second half to the first half. Let  $\rho' \in \{0, 1\}^n$  be the partial input consistent with  $x$  such that  $r(\rho') = B$ . Since  $|\rho'| = |B| \leq |\rho| < C_0(f)$ , we know that  $\rho'$  does not certify  $f(x) = 0$ . This means there is some 1-certificate  $\sigma \in U$  consistent with  $\rho'$  and such that  $r(\sigma) \not\subseteq r(\rho')$ . Our goal becomes to use  $\sigma$  to modify  $z$  in positions outside  $r(\rho)$  to obtain a  $z'$  such that  $g(z') = 1$ , which would show that  $\rho$  is not a  $\bar{1}$ -certificate, concluding the proof. Indeed, starting with  $z = (x, 0^n)$  we can modify the first half  $x$  to contain  $\sigma$ , and we can modify the bits  $r(\sigma) \setminus B \neq \emptyset$  in the second half  $0^n$  so that the positions  $r(\sigma)$  (in the second half) have odd parity.  $\square$

## 4.3 Construction II $\Rightarrow$ III

**Given:** A partial function  $f: \{0, 1\}^n \rightarrow \{0, 1, *\}$  and  $x \in f^{-1}(*)$ .

**Construct:** An intersecting hypergraph  $G = (V, E)$  with  $|V| = 2n + 2$  and  $r(G) = C(f) + 1$  and a colouring  $c: V \rightarrow \{0, 1\}$  such that every  $c$ -monochromatic hitting set has size at least  $\min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\}$ .

For each  $i \in [n]$ , we introduce two vertices  $v_{i,0}$  and  $v_{i,1}$  into  $V$ . We also have two special vertices, which we denote  $u_0$  and  $u_1$ . For each 0-certificate  $\rho \in \{0, 1, *\}^n$  of size  $|\rho| \leq C(f)$ , we construct an edge  $S_\rho$ , as follows. For each  $i \in [n]$ , if  $\rho_i = 0$  we place  $v_{i,0}$  in  $S_\rho$ , and if  $\rho_i = 1$  we place  $v_{i,1}$  in  $S_\rho$ . We also place  $u_0$  in  $S_\rho$ . Then  $|S_\rho| = |\rho| + 1 \leq C(f) + 1$ .

For each 1-certificate  $\rho$  of size  $|\rho| \leq C(f)$ , we construct an edge  $S_\rho$  slightly differently. Essentially, we negate the bits of  $\rho$  before creating the edge out of  $\rho$ . So if  $\rho_i = 0$  we place  $v_{i,1}$  in  $S_\rho$  and if  $\rho_i = 1$



we place  $v_{i,0}$  in  $S_\rho$ . We also place  $u_1$  in  $S_\rho$ . Together, the edges coming from 0- and 1-certificates constitute all the edges in  $E$ . This defines  $G = (V, E)$ .

Note that  $r(G) = C(f) + 1$ . Additionally,  $G$  is intersecting. To see this, note that if  $S_\rho$  and  $S_{\rho'}$  are two edges of  $G$ , then there are three options: if  $\rho$  and  $\rho'$  are both 0-certificates, they share  $u_0$ ; if  $\rho$  and  $\rho'$  are both 1-certificates, they share  $u_1$ ; and if  $\rho$  and  $\rho'$  are certificates of opposite types, then they must contradict each other at some index, meaning that  $\rho_i = 0$  and  $\rho'_i = 1$  (or vice versa) for some  $i \in [n]$ . In this last case,  $S_\rho$  and  $S_{\rho'}$  either both contain  $v_{i,0}$  or both contain  $v_{i,1}$ . In all cases,  $S_\rho$  and  $S_{\rho'}$  have a non-empty intersection.

We now define the colouring  $c: V \rightarrow \{0, 1\}$  based on the input  $x \in f^{-1}(*)$ . We do so by setting  $c(v_{i,x_i}) = 0$ ,  $c(v_{i,1-x_i}) = 1$ ,  $c(u_0) = 0$ , and  $c(u_1) = 1$ . It remains to prove the following claim.

**Claim 14.** *If  $H$  is a  $c$ -monochromatic hitting set for  $G$ , then  $|H| \geq \min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\}$ .*

*Proof.* If  $H$  uses the colour 1, then it does not contain  $u_0$ ; since it is a hitting set, it must intersect  $S_\rho$  for each short 0-certificate  $\rho$  in some vertex  $v_{i,b}$  (where  $i \in [n]$  and  $b \in \{0, 1\}$ ). Since  $H$  is monochromatic with colour 1, we must have  $b = 1 - x_i$ . Since  $v_{i,1-x_i} \in S_\rho$ , we must have  $\rho_i = 1 - x_i$ . In other words, the hitting set  $H$  must define a set of indices in  $[n]$  such that for each short 0-certificate  $\rho$  of  $f$ , there is some index  $i$  in this set on which  $\rho$  contradicts  $x$ . Since each 0-input to  $f$  contains a short 0-certificate (of length at most  $C(f)$ ), we conclude that this set of indices used by  $H$  is such that each 0-input to  $f$  conflicts with  $x$  in one of those indices. This means that we can construct a  $\bar{0}$ -certificate by reading these indices in the string  $x$ ; thus  $|H| \geq C_{\bar{0}}(f, x)$ .

Alternatively, suppose that  $H$  uses the colour 0. Then it does not contain  $u_1$ , and must intersect each  $S_\rho$  for a short 1-certificate  $\rho$  of  $f$  in a vertex  $v_{i,b}$ . Since  $H$  uses the colour 0, we must have  $b = x_i$ , and since  $v_{i,x_i} \in S_\rho$ , we must have  $\rho_i = 1 - x_i$ . As before, this implies that  $H$  defines a set of indices such that each short 1-certificate of  $f$  contradicts  $x$  on one of these indices; hence we can get a  $\bar{1}$ -certificate by reading those indices in  $x$ , which implies that  $|H| \geq C_{\bar{1}}(f, x)$ .  $\square$

#### 4.4 Construction III $\Rightarrow$ II

**Given:** An intersecting hypergraph  $G = (V, E)$  and a colouring  $c: V \rightarrow \{0, 1\}$  such that every  $c$ -monochromatic hitting set has size at least  $h > r(G)$ .

**Construct:** A partial boolean function  $f: \{0, 1\}^V \rightarrow \{0, 1, *\}$  and an input  $x \in f^{-1}(*)$  such that  $C(f) \leq r(G)$  and  $\min\{C_{\bar{0}}(f, x), C_{\bar{1}}(f, x)\} \geq h$ .

We define  $f$  on  $n = |V|$  bits so that an input to  $f$  is interpreted as a colouring of  $V$ . We define  $f(z) := 0$  if the colouring  $z$  contains a monochromatic edge of colour 0, and we define  $f(z) := 1$  if  $z$  contains a monochromatic edge of colour 1. Note that both cases cannot hold, because  $G$  is intersecting. If neither of these cases holds, we define  $f(z) := *$ .

To certify that  $f(z) = 0$  or that  $f(z) = 1$ , we can just read a monochromatic edge in  $z$ ; this only uses  $r(G)$  bits in the worst case over 0- or 1-inputs  $z$ , so  $C(f) \leq r(G)$ .

Next, consider the input  $x$  to  $f$  which is defined by the colouring  $c$ . Since any monochromatic edge is a monochromatic hitting set (since  $G$  is intersecting, so every edge is a hitting set), and since the minimum monochromatic hitting set in  $c$  has size  $h > r(G)$ , we conclude that  $c$  does not have a monochromatic edge, and hence  $f(x) = *$ . Observe that a certificate that  $x$  is not a 0-input is a proof that there is no 0-monochromatic edge in  $c$ , and such a proof must necessarily read a 1-monochromatic hitting set in  $c$ ; hence  $C_{\bar{0}}(f, x) \geq h$ . Similarly, we have  $C_{\bar{1}}(f, x) \geq h$ .

*Remark 15.* We note that  $f$  is *monotone* by construction: flipping any bit in an input  $z$  from 0 to 1 can only change  $f(z)$  from 0 to \* or 1, or from \* to 1. In particular, this means that we can transform any solution to **II** into a monotone one via the steps **II** $\Rightarrow$ **III** $\Rightarrow$ **II**.

## 5 Application: Approximate degree vs. certificate complexity

Finally, we prove [Corollary 5](#), which states that there exists a total function  $f$  with  $C(f) \geq \tilde{\Omega}(\widetilde{\deg}(f)^3)$ . Let us quickly recall the definition of the  $\epsilon$ -approximate degree  $\widetilde{\deg}_\epsilon(f)$  of an  $n$ -bit boolean function  $f$ : it equals the least degree of an  $n$ -variate polynomial  $p: \mathbb{R}^n \rightarrow \mathbb{R}$  such that  $p(x) \in f(x) \pm \epsilon$  for every boolean input  $x \in \{0, 1\}^n$ . We also set  $\widetilde{\deg}(f) := \widetilde{\deg}_{1/3}(f)$ .

**Proof of Corollary 5.** By applying the construction **II** $\Rightarrow$ **I** ([Section 4.1](#)) to our HEX function ([Section 3](#)), we get a total  $g$  with  $C_0(g) \geq \tilde{\Omega}(\text{UC}_1(g)^{1.5})$ . All we have to show is that  $g$  also has

$$\widetilde{\deg}(g) \leq \tilde{O}(\sqrt{\text{UC}_1(g)}). \quad (1)$$

Lets examine the function constructed by **II** $\Rightarrow$ **I** using the notation in that proof. This proof starts out with an original  $n$ -bit function  $f$  (namely,  $\text{HEX}_{\sqrt{n}}$ ) and it defines from it a new function  $g$  on  $O(n^2 \log^2 n)$  bits using the cheat sheet framework. An input to  $g$  consists of  $k := \log n$  inputs to  $f$  and an array of size  $n$ , where each cell of the array is of size  $k\ell$ , where  $\ell \leq 2C(f) \log n$  is the number of bits needed to specify a certificate of  $f$ . In a 1-input to  $g$ , the correct cell, which is cell  $s_z$ , is supposed to contain  $k$  certificates for the  $k$  instances of  $f$ . We did not specify how the certificates would be described since the construction **II** $\Rightarrow$ **I** applies to a general function  $f$ , but now let us describe them precisely for  $f = \text{HEX}$ . Here, a convenient 0-certificate is a list of adjacent 0-entries that starts from the left and ends on the right. For a 1-certificate we can have a similar list that starts at the top and ends at the bottom. Let us modify our function  $g$  to require that the certificates are presented in exactly this format.

Now for any cell  $c$ , consider the boolean function  $g_c$  that on an input  $z$  to  $g$  evaluates to 1 if  $g(z) = 1$  and additionally that cell  $c$  is the one pointed to by  $z$  (that is,  $s_z = c$ ). We will show that this boolean function has an approximating polynomial of degree  $\tilde{O}(\sqrt{\text{UC}_1(g)})$ .

To check if cell  $c$  is the one pointed to by the  $\log n$  copies of  $f$ , we first need to check that the certificates contained in  $c$  are valid certificates for the  $\log n$  instances of  $f$ , and that  $\log n$   $f$ -outputs of these certificates, when interpreted as a number is indeed  $c$ . First we claim that checking if a certificate for a particular  $f$  is valid can be done with an approximating polynomial of degree  $\tilde{O}(\sqrt{\text{UC}_1(g)})$ . Let us do this for 0-certificates, and the construction for 1-certificates is similar. Each 0-certificate for an instance of  $f$  contains  $C(f)$  many HEX-matrix entries that are adjacent, all having the value 0, and starting at the left and ending at the right. Checking if two adjacent entries in the list correspond to adjacent matrix entries requires only  $O(\log n)$  queries by a deterministic query algorithm (decision tree). There are  $O(n)$  such checks to be made. Checking if a matrix entry in the list is 0 requires  $O(\log n)$  queries as well. There are  $O(n)$  such checks to be made. And finally checking that the first and last entry of the list are on the left and right require  $O(\log n)$  queries. In total we have to make  $O(n)$  checks, each of which cost  $O(\log n)$  queries. Equivalently, we want to compute the logical AND of  $O(n)$  many query algorithms, each of which has query complexity  $O(\log n)$ .

A deterministic query algorithm of  $O(\log n)$  queries can be converted to an exact polynomial of degree  $O(\log n)$ . Nisan and Szegedy [[NW95](#)] showed that there is a polynomial of degree  $O(\sqrt{n})$  to approximate the  $n$ -bit AND function. Composing this polynomial with a  $O(\log n)$ -degree polynomials for the individual checks gives us an approximating polynomial of degree  $\tilde{O}(\sqrt{n})$  for checking if

a particular certificate for  $f$  is valid. Since there are  $\log n$  certificates to be checked, checking all of them does not increase the degree by more than a log factor. Once we have checked if all the  $f$  certificates are valid, we know the outputs and can check if this equals  $c$ . Thus we have an approximating polynomial for  $g_c$  of degree  $\tilde{O}(\sqrt{n})$ .

Now that we know that  $g_c$  has an approximating polynomial of degree  $\tilde{O}(\sqrt{n})$ , we can construct one for  $g$  from such polynomials. First we boost the approximation accuracy of the polynomials we constructed to have error  $1/3n$ , which only increases the degree by a log factor. Then we observe that  $g(z) = 1$  if and only if one of the  $g_c(z)$  functions evaluate to 1, and furthermore, no more than one of them can evaluate to 1 since these are unambiguous certificates. So we get an approximate polynomial for  $g$  by simply summing up the polynomials for all  $g_c$ . Since each polynomial had error  $1/3n$ , the resulting polynomial has error at most  $1/3$ . The degree has not increased, and hence we have an approximating polynomial for  $g$  of degree  $\tilde{O}(\sqrt{n}) = \tilde{O}(\sqrt{\text{UC}_1(f)})$ . This proves (1).

## Acknowledgements

Thanks to Ryan Alweiss, Harry Buhrman, Nati Linial, and Mario Szegedy for their thoughts on [Puzzle III](#). Thanks to Thomas Watson for many discussions about Hex and complexity classes.

## References

- [ABK16] Scott Aaronson, Shalev Ben-David, and Robin Kothari. Separations in query complexity using cheat sheets. In *Proceedings of the 48th Symposium on Theory of Computing (STOC)*, pages 863–876. ACM, 2016. [doi:10.1145/2897518.2897644](https://doi.org/10.1145/2897518.2897644).
- [ABK<sup>+</sup>21] Scott Aaronson, Shalev Ben-David, Robin Kothari, Shravas Rao, and Avishay Tal. Degree vs. approximate degree and quantum implications of Huang’s sensitivity theorem. In *Proceedings of the 53rd Symposium on Theory of Computing (STOC)*, 2021. To appear. [arXiv:2010.12629](https://arxiv.org/abs/2010.12629).
- [Ama14] Kazuyuki Amano. Some improved bounds on communication complexity via new decomposition of cliques. *Discrete Applied Mathematics*, 166(0):249–254, 2014. [doi:10.1016/j.dam.2013.09.015](https://doi.org/10.1016/j.dam.2013.09.015).
- [BHT17] Shalev Ben-David, Pooya Hatami, and Avishay Tal. Low-sensitivity functions from unambiguous certificates. In *8th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 67, pages 28:1–28:23. Schloss Dagstuhl, 2017. [doi:10.4230/LIPIcs.ITCS.2017.28](https://doi.org/10.4230/LIPIcs.ITCS.2017.28).
- [BLMP18] Nicolas Bousquet, Aurélie Lagoutte, Frédéric Maffray, and Lucas Pastor. Decomposition techniques applied to the clique-stable set separation problem. *Discrete Mathematics*, 341(5):1492–1501, 2018. [doi:10.1016/j.disc.2017.10.014](https://doi.org/10.1016/j.disc.2017.10.014).
- [BLT14] Nicolas Bousquet, Aurélie Lagoutte, and Stéphan Thomassé. Clique versus independent set. *European Journal of Combinatorics*, 40(0):73–92, 2014. [doi:10.1016/j.ejc.2014.02.003](https://doi.org/10.1016/j.ejc.2014.02.003).
- [CS21] Maria Chudnovsky and Paul Seymour. Subdivided claws and the clique-stable set separation property. *2019-20 MATRIX Annals*, pages 483–487, 2021. [doi:10.1007/978-3-030-62497-2\\_29](https://doi.org/10.1007/978-3-030-62497-2_29).

- [GJPW18] Mika Göös, T. S. Jayram, Toniann Pitassi, and Thomas Watson. Randomized communication vs. partition number. *ACM Transactions on Computation Theory*, 10(1):4:1–4:20, 2018. doi:[10.1145/3170711](https://doi.org/10.1145/3170711).
- [GLM<sup>+</sup>16] Mika Göös, Shachar Lovett, Raghu Meka, Thomas Watson, and David Zuckerman. Rectangles are nonnegative juntas. *SIAM Journal on Computing*, 45(5):1835–1869, 2016. doi:[10.1137/15M103145X](https://doi.org/10.1137/15M103145X).
- [Göö15] Mika Göös. Lower bounds for clique vs. independent set. In *Proceedings of the 56th Symposium on Foundations of Computer Science (FOCS)*, pages 1066–1076. IEEE, 2015. doi:[10.1109/FOCS.2015.69](https://doi.org/10.1109/FOCS.2015.69).
- [GPW18] Mika Göös, Toniann Pitassi, and Thomas Watson. Deterministic communication vs. partition number. *SIAM Journal on Computing*, 47(6):2435–2450, 2018. doi:[10.1137/16M1059369](https://doi.org/10.1137/16M1059369).
- [HS12] Hao Huang and Benny Sudakov. A counterexample to the Alon–Saks–Seymour conjecture and related problems. *Combinatorica*, 32(2):205–219, 2012. doi:[10.1007/s00493-012-2746-4](https://doi.org/10.1007/s00493-012-2746-4).
- [LT16] Aurélie Lagoutte and Théophile Trunck. Clique–Stable Set separation in perfect graphs with no balanced skew-partitions. *Discrete Mathematics*, 339(6):1809–1825, 2016. doi:[10.1016/j.disc.2016.02.005](https://doi.org/10.1016/j.disc.2016.02.005).
- [NW95] Noam Nisan and Avi Wigderson. On rank vs. communication complexity. *Combinatorica*, 15(4):557–565, 1995. doi:[10.1007/BF01192527](https://doi.org/10.1007/BF01192527).
- [Raz11] Ran Raz. How to fool people to work on circuit lower bounds, 2011. Seminar talk. URL: <https://youtu.be/nsQzS3IOS6Y>.
- [SA15] Manami Shigeta and Kazuyuki Amano. Ordered biclique partitions and communication complexity problems. *Discrete Applied Mathematics*, 184:248–252, 2015. doi:[10.1016/j.dam.2014.10.029](https://doi.org/10.1016/j.dam.2014.10.029).
- [Yan91] Mihalis Yannakakis. Expressing combinatorial optimization problems by linear programs. *Journal of Computer and System Sciences*, 43(3):441–466, 1991. doi:[10.1016/0022-0000\(91\)90024-Y](https://doi.org/10.1016/0022-0000(91)90024-Y).