The Journal of Symbolic Logic
Volume 85, Number 2, June 2020

ON OBDD-BASED ALGORITHMS AND PROOF SYSTEMS THAT DYNAMICALLY CHANGE THE ORDER OF VARIABLES

DMITRY ITSYKSON, ALEXANDER KNOP, ANDREI ROMASHCHENKO, AND DMITRY SOKOLOV

Abstract. In 2004 Atserias, Kolaitis, and Vardi proposed OBDD-based propositional proof systems that prove unsatisfiability of a CNF formula by deduction of an identically false OBDD from OBDDs representing clauses of the initial formula. All OBDDs in such proofs have the same order of variables. We initiate the study of OBDD based proof systems that additionally contain a rule that allows changing the order in OBDDs. At first we consider a proof system OBDD($\wedge$, reordering) that uses the conjunction (join) rule and the rule that allows changing the order. We exponentially separate this proof system from OBDD($\wedge$) proof system that uses only the conjunction rule. We prove exponential lower bounds on the size of OBDD($\wedge$, reordering) refutations of Tseitin formulas and the pigeonhole principle. The first lower bound was previously unknown even for OBDD($\wedge$) proofs and the second one extends the result of Tveretina et al. from OBDD($\wedge$) to OBDD($\wedge$, reordering).

In 2001 Aguirre and Vardi proposed an approach to the propositional satisfiability problem based on OBDDs and symbolic quantifier elimination (we denote algorithms based on this approach as OBDD($\wedge, \exists$) algorithms). We augment these algorithms with the operation of reordering of variables and call the new scheme OBDD($\wedge, \exists$, reordering) algorithms. We notice that there exists an OBDD($\wedge, \exists$) algorithm that solves satisfiable and unsatisfiable Tseitin formulas in polynomial time (a standard example of a hard system of linear equations over $\mathbb{F}_2$), but we show that there are formulas representing systems of linear equations over $\mathbb{F}_2$ that are hard for OBDD($\wedge, \exists$, reordering) algorithms. Our hard instances are satisfiable formulas representing systems of linear equations over $\mathbb{F}_2$ that correspond to checksum matrices of error correcting codes.

§1. Introduction.

An ordered binary decision diagram (OBDD) represents a Boolean function. A Boolean function is represented as a branching program with two sinks such that on every path from the source to a sink variables appear in the same order. This restriction on the order of variables allows handling the diagrams (compute binary Boolean operations on diagrams, compute projections, or test satisfiability) efficiently [22].

Atserias, Kolaitis, and Vardi [3] proposed an OBDD-based proof system. This system is meant to prove that a given CNF formula is unsatisfiable. For some order of variables π we represent clauses of the input formula as a π-ordered BDD; we

Received January 27, 2019.
2020 *Mathematics Subject Classification.* 68Q11, 68R07, 05C48, 94B05.
Key words and phrases. communication complexity, error correcting codes, OBDD, proof complexity, satisfiability algorithms.

The present paper is an expansion of an earlier conference paper [17]. It includes the contents of [17] augmented with all the proofs that were omitted in the conference version. In addition, the paper contains a new deterministic construction of a hard example for OBDD($\wedge, \exists$) algorithms.

0022-4812/20/8502-0006
DOI:10.1017/jsl.2019.53

may derive a new OBDD applying either the *conjunction* rule or the *weakening* rule. (The authors of [3] supplied the system with one more rule—the *projection*, which derives $\exists x D$ from D; we consider this rule as a special case of the weakening rule, so we do not need to allow it explicitly.) A *proof* in this system is a derivation of an OBDD that represents the constant false function. We refer to this proof system as the OBDD($\wedge$, weakening) proof system. The OBDD($\wedge$, weakening) proof system simulates Cutting Planes with unary coefficients [3] and thus it is stronger than Resolution [9]. In addition, formulas representing unsatisfiable systems of linear equations over $\mathbb{F}_2$ have short proofs in this proof system [3], while linear systems are hard for Resolution. This observation motivates the study of the OBDD-based algorithms for satisfiability problem (notice that the popular DPLL and CDCL algorithms for satisfiability problem correspond to tree-like and DAG-like Resolutions).

Several strong lower bounds are known for different versions of OBDD proof systems. Segerlind [25] proved an exponential lower bound for the tree-like version of OBDD($\wedge$, weakening) proof system using the communication complexity technique proposed in [16]. Krajicek [19] proved an exponential lower bound for the DAG-like version of it using monotone feasible interpolation [4, 18].

Several papers study the OBDD-based proof system that has only one inference rule—the conjunction rule (we refer to this system as OBDD($\wedge$) proof system); the proof search in this restricted proof system seems to be easier than in the general proof system. Groote and Zantema [13] showed that there is a formula ϕ (not in CNF) such that the Tseitin transformation of this formula does not have a short resolution refutations but ϕ has short OBDD($\wedge$) refutations. Buss et al. [5] showed that OBDD($\wedge$) is not polynomially simulated by Cutting Planes, and thus by Resolution. Tveretina, Sinz, and Zantema [27] proved the lower bound $2^{\Omega(n)}$ for the pigeonhole principle $\mathtt{PHP}_n^{n+1}$ in the OBDD($\wedge$) proof system. Friedman and Xu [11] proved an exponential lower bound for the complexity of random unsatisfiable 3CNF formulas in restricted versions of OBDD($\wedge$) proof systems (with a fixed order of the variables).

An interesting approach of incorporating a weak analog of the weakening rule to solving propositional satisfiability was suggested by Aguirre and Vardi [1]. They proposed an algorithm that chooses some order π on the variables of the input CNF formula F and creates the current π-ordered BDD D that initially represents the constant true function, and a set of clauses S that initially consists of all clauses of the formula F. Then the algorithm applies the following operations in an arbitrary order:

Conjunction (or join): choose a clause $C \in S$, delete it from S and replace D by the π-ordered BDD representing $C \wedge D$;

Projection (or $\exists$-elimination): choose a variable x that has no occurrence in the clauses from S and replace D by the π-ordered BDD representing $\exists x D$.

When S becomes empty, the algorithm stops and reports "unsatisfiable" if D represents the constant false function and "satisfiable" otherwise. Every particular instance of this algorithm uses its own strategies to choose an order of variables π and an order of application of the operations. We refer to these algorithms as OBDD($\wedge, \exists$) algorithms.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Pan and Vardi [24] investigated some specific strategies and compared them with DPLL based SAT solvers and SAT solvers based on OBDDs without quantifier eliminations (we call them OBDD($\wedge$) algorithms). Experiments showed in particular that OBDD($\wedge, \exists$) algorithms are faster than OBDD($\wedge$) algorithms [24]. A result of [7] implies that an OBDD($\wedge, \exists$) algorithm can solve $\mathtt{PHP}_n^{n+1}$ in polynomial time. The lower bounds for the OBDD($\wedge$, weakening) proof systems mentioned above imply the same lower bounds for the OBDD($\wedge, \exists$) algorithms.

1.1. Statement of the problem. It is known that changing the order of the variables in an OBDD can be performed in time polynomial in the sizes of the input and the output [22]. So it seems to be unnecessarily restrictive to use the same order of variables in all OBDDs in the proof. Hence, we propose to strengthen the OBDD proof systems with a supplementary rule that dynamically reorders the variables in OBDDs.

In OBDD proofs, reordering rule may be applied to an arbitrary OBDD from the proof but the conjunction rule may be applied only to OBDDs with the same order since the verification of the application of the conjunction to OBDDs in different orders is hard (this problem is coNP-complete, see [23, Lemma 8.14]).

The first aim of this paper is to prove lower bounds for the OBDD-based algorithms and proof systems that use the reordering rule. The second aim is to show that reordering of the variables is really useful; we give examples of formulas that are hard for the OBDD-based proof systems without reordering and easy with reordering.

1.2. Our results. In Section 3 we study the OBDD($\wedge$, reordering) proof system. It proves exponential lower bounds on size of OBDD($\wedge$, reordering) refutations of the pigeonhole principle $\mathtt{PHP}_n^{n+1}$ and Tseitin formulas based on constant-degree algebraic expanders. The lower bound on the pigeonhole principle extends the result of Tveretina et al. [27] from OBDD($\wedge$) refutations to OBDD($\wedge$, reordering) refutations. (Besides, we believe that our argument is simpler than the proof in [27].) The result for Tseitin formulas, to the best of our knowledge, was not known even for the more restrictive OBDD($\wedge$) refutations. In both arguments we use the same strategy:

- At first step, we prove an exponential lower bound on the size of the OBDD-representation of an appropriate satisfiable formula. Assume that the original unsatisfiable formula is minimally unsatisfiable. Roughly speaking, the satisfiable formula under consideration is similar to the original unsatisfiable formula with several canceled clauses. For example, for the pigeonhole principle the appropriate satisfiable formula would be $\mathtt{PHP}_n^n$; for the Tseitin formulas such an appropriate formula is a satisfiable Tseitin formula. This part of the proof is quite cumbersome but it involves only rather elementary techniques of lower bounds for OBDD.
- Consider the last step of the refutation. It consists in the conjunction of F_1 and F_2 in the same order π. Our goal is to prove that at least one of F_1 and F_2 has an exponential size. Both F_1 and F_2 are satisfiable and they are conjunctions of different sets of clauses of the initial formulas. The idea is to construct partial substitutions ρ_1 and ρ_2 with the same support such that the formula

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

$F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is isomorphic to the satisfiable formula from the first step. Then any OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has exponential size. Hence, the size of either $F_1|_{\rho_1}$ or $F_2|_{\rho_2}$ is large whence the size of either F_1 or F_2 is large.

In Section 4 we construct an example of a family of formulas that are easy for the OBDD($\wedge$, reordering) proof system but hard for the OBDD($\wedge$) proof system.

Finally, Section 5 studies OBDD($\wedge, \exists$, reordering) algorithms. At first, we notice that there exists an OBDD($\wedge, \exists$) algorithm that solves satisfiable and unsatisfiable Tseitin formulas in polynomial time. In contrast, we show that there exist formulas representing systems of linear equations over $\mathbb{F}_2$ that are hard for OBDD($\wedge, \exists$, reordering) algorithms. More specifically, we describe a construction of a family of satisfiable formulas F_n on n variables in $O(1)$-CNF such that every OBDD($\wedge, \exists$, reordering) algorithm runs at least $2^{\Omega(n)}$ steps on F_n.

The plan of the proof is as follows.

- We prove that if $C \subseteq \{0,1\}^n$ is the set of codewords of a list-decodable code that allows to correct $\frac{2}{3}n$ of erasures, then every OBDD that represents the characteristic function χ_C of C has a big size (this size proved to be close to the number of all codewords in the code). Moreover, this property holds even for the projections of χ_C onto any $(1 - \frac{1}{6})n$ coordinates. Notice that a similar result was known for error-correcting codes with large enough minimal distance (see for example the paper of Duris et al. [10]). Guruswami [14] showed that the codes with large enough minimal distance are erasure list-decodable but the opposite statement does not hold.
- In Section 6.1, we give a randomized construction of the required linear code. This construction is based on random checksum matrices. We use the codes of Gallager [12] that contain $O(1)$ ones per row. We prove that such a random code with a high probability enjoys the following expansion property: every $\frac{1}{6}n$ columns of the matrix contain ones in almost all rows. And in Section 6.2 we give an explicit construction of the required linear code.
- We consider the execution of an OBDD($\wedge, \exists$, reordering) algorithm on CNF formula that corresponds to the CNF representation of the checksum matrix of the code. We study two cases:
  1. the algorithm applies the projection rule less than $\frac{n}{6}$ times;
  2. the projection rule is applied at least $\frac{n}{6}$ times.

  In the first case, we focus on the OBDD at the end of the execution of the algorithm; its size should be exponential due to the properties of the code. In the second case, we consider the first moment in the computational process when we apply exactly $\frac{n}{6}$ projection operations. By the expansion property, the OBDDD is a projection of almost the entire formula, thus its size should be close to the size of the OBDD of the characteristic function of the code. That is, the size of D should be large enough.

As we mentioned above, the previously known lower bounds for tree-like and DAG-like OBDD($\wedge$, weakening) proofs imply lower bounds for OBDD($\wedge, \exists$) algorithms. So, what is new in our results comparative to the lower bounds proven by Segerlind [26] and Krajíček [19]? First of all, our lower bound also works for the reordering operation. The second advantage of our construction is that we come up with quite a natural class of formulas (our formulas represent linear systems of equations that define some error correcting codes), while the constructions in [19, 26] seem to be rather artificial. Further, we prove the lower bound $2^{\Omega(n)}$ for a formula with n variables, whereas the previously known lower bounds are of the type $2^{n^{\varepsilon}}$ (for some $\varepsilon < 1/5$). Besides, we proposed a new technique that might be applicable for other classes of formulas.

§2. Preliminaries.

2.1. Ordered binary decision diagrams. A binary decision diagram (BDD) is a data structure that is used to represent Boolean functions. Let $\Gamma = \{x_1, \dots, x_n\}$ be a set of propositional variables. A binary decision diagram is a directed acyclic graph with one source. Every vertex of the graph is labeled by a variable from Γ or by a constant 0 or 1. If a vertex is labeled by a constant, then it is a sink (has out-degree 0). If a vertex is labeled by a variable, then it has exactly two outgoing edges: one edge is labeled by 0 and the other edge is labeled by 1. Every binary decision diagram defines a Boolean function $\{0,1\}^n \to \{0,1\}$. The value of the function for given values of $x_1, \dots, x_n$ is computed as follows: we start a path at the source and on every step we go along the edge that corresponds to the value of the variable at the current vertex. Every such path reaches a sink labeled by the constant; this constant is the value of the function. The size of a BDD is equal to the number of vertices in the graph.

Let π be a permutation of the set $\{1, \dots, n\}$. A π-ordered binary decision diagram (π-OBDD) is a binary decision diagram such that on every path from the source to a sink every variable has at most one occurrence and variable $x_{\pi(i)}$ cannot appear before $x_{\pi(j)}$ if $i > j$. An ordered binary decision diagram (OBDD) is a π-ordered binary decision diagram for some permutation π.

OBDDs have the following nice property: for every order of variables every Boolean function has a unique minimal OBDD. For a given order π, the minimal π-OBDD of a function f may be constructed in polynomial time from any π-OBDD for the same function [22]. There are also known polynomial-time algorithms that efficiently perform the operations of conjunction, negation, disjunction, and projection (elimination of the existential quantifier) to π-ordered BDDs [22] (see [29] for a good introduction to OBDDs). Finally, there is an algorithm running in time polynomial in the size of the input and the output that gets as an input a π-ordered diagram A, a permutation ρ and returns the minimal ρ-ordered diagram that represents the same function as A [22].

2.2. Communication complexity. In order to prove lower bounds on the size of an OBDD representation of a function we will use communication complexity.

Let $f : \{0,1\}^n \to \{0,1\}$ be some function and $\Pi = (\Pi_0, \Pi_1)$ be a partition of $[n]$ i.e., $\Pi_0 \cup \Pi_1 = [n]$ and $\Pi_0 \cap \Pi_1 = \emptyset$. Two players Alice and Bob want to compute $f(x)$ for some x. However, Alice knows only bits of x with indices from Π_0 and Bob knows bits of x with indices from Π_1. In order to compute the value they can use a two-sided communication channel. They agreed in advance about a protocol; on each step of the protocol one of them sends a bit string to the other, at the end of the protocol both Alice and Bob should know $f(x)$. The cost of the protocol is the

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

maximal number of bits they sent to each other. The communication complexity of f with respect to the partition Π is equal to the minimal cost of the protocols for f (for the formal definition see [20]).

It is well-known that there is a connection between communication complexity and the minimal size of an OBDD representation.

Lemma 2.1 ([20]). *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function, $\Pi = (\Pi_0, \Pi_1)$ be a partition of $[n]$. Then for any order π such that elements of Π_0 precede elements of Π_1 in this order, any π-*OBDD *representation of f has size at least 2^{C-2} where C is equal to the communication complexity of f with respect to Π.*

Proof. Let us fix some π-OBDD representation D of f. Let us consider the path p in D from the source to a sink corresponding to the Alice and Bob's input x.

Since D is a π-OBDD, all the variables on the path appear according to the order π. This implies that the path can be split into 2 blocks so that in each block either all variables are known to Alice or all variables are known to Bob.

Let us show that Alice and Bob can find the sink of the path using $\lceil \log |D| \rceil + 1$ bits of communication. They do it in 2 rounds; starting from the source (which is known to both of them) they follow the path, and on each round one of the players sends to the other the name of the vertex where the current block ends; at the end, when one player reaches the sink, this player communicate the name of the sink to the other one. The value $f(x)$ is determined by the sink vertex. Hence, the communication complexity of f is at most $\lceil \log |D| \rceil + 1$. ⊣

In what follows, we will need to prove lower bounds on communication complexity. First of all we recall the technique of combinatorial rectangles. Let $f : X \times Y \to \{0,1\}$ be a function. We say that $R = A \times B$ is a 1-monochromatic rectangle for f iff $A \subseteq X$, $B \subseteq Y$, and for every $a \in A$ and $b \in B$, $f(a,b) = 1$. We say that a sequence of rectangles R_1, ..., R_ℓ is a partition of $f^{-1}(1)$ into 1-monochromatic rectangles iff

- $R_i \cap R_j = \emptyset$ for every $i,j \in [\ell]$ and
- for every $x \in X$ and $y \in Y$, $f(x,y) = 1$ iff $(x,y) \in R_i$, for some i.

Lemma 2.2 ([20]). *Let Π be a partition of $[n]$ and $f : \{0,1\}^{\Pi_0} \times \{0,1\}^{\Pi_i} \to \{0,1\}$ be a Boolean function. If any partition of $f^{-1}(1)$ into 1-monochromatic rectangles has at least T rectangles, then the communication complexity of f is at least $\log(T+1)$.*

Lemma 2.3. *Let $f : \{0,1\}^n \to \{0,1\}$ be a Boolean function and Π be a partition of $[n]$. If there are substitutions ρ_1, ..., ρ_ℓ to the variables $\{x_i \mid i \in \Pi_0\}$ such that*

- *for every $i \in [\ell]$ the function $f|_{\rho_i}$ is satisfiable (i.e., it is not identically zero) and*
- *for every $i \neq j \in [\ell]$ the satisfying assignments of $f|_{\rho_i}$ and $f|_{\rho_j}$ are disjoint.*

Then the communication complexity of f with respect to Π is at least $\lfloor \log \ell \rfloor + 1$.

To prove this lemma we need the following definition and the following lemma. The equality function $\mathtt{EQ}_n : \{0,1\}^{2n} \to \{0,1\}$ is a Boolean function such that $\mathtt{EQ}_n(x_1, \dots, x_n, y_1, \dots, y_n) = 1$ iff $x_i = y_i$ for all $i \in [n]$.

Lemma 2.4 ([20]). *Let $\Pi = ([1,n],[n+1,2n])$ be a partition of $[2n]$. The communication complexity of $\mathtt{EQ}_n$ with respect to the partition Π is equal to $n+1$.*

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Proof of Lemma 2.3. Let us assume that there is a communication protocol $\mathcal{P}$ for f of cost S. We are going to show that there is a protocol for $\mathtt{EQ}_{\lfloor \log \ell \rfloor}$ of cost S and hence, by Lemma 2.4, $S \geq \lfloor \log \ell \rfloor + 1$.

Let us assume that Alice knows $x \in \{0,1\}^{\lfloor \log \ell \rfloor}$ and Bob knows $y \in \{0,1\}^{\lfloor \log \ell \rfloor}$ and they wish to check whether $\mathtt{EQ}_{\lfloor \log \ell \rfloor}(x,y) = 1$, i.e., $x = y$.

Let y be the binary representation of $j \in [\ell]$ and x be the binary representation of $i \in [\ell]$. In order to check whether $x = y$, Bob finds a satisfying assignment τ for $f|_{\rho_j}$. After that they run the protocol $\mathcal{P}$ for ρ_i and τ. Note that if $x = y$, then $\rho_i = \rho_j$ and τ satisfies $f|_{\rho_i} = f|_{\rho_j}$, i.e., $f|_{\rho_i \tau} = 1$; otherwise $i \neq j$ and sets of satisfying assignments of $f|_{\rho_i}$ and $f|_{\rho_j}$ are disjoint, hence, $f|_{\rho_i \tau} = 0$. ⊣

2.3. OBDD **proofs.** If F is a formula in CNF, we say that the sequence $D_1, D_2, \dots, D_t$ is an OBDD derivation of D_t from F if for all $1 \leq i \leq t$, D_i is an OBDD that represents a clause of F or can be obtained from the previous D_j's by one of the following inference rules:

Conjunction or join: D_i is a π-ordered OBDD, that represents $D_k \wedge D_\ell$ for $1 \leq \ell, k < i$, where D_k, D_ℓ have the same order π;

Weakening: there exists a $j < i$ such that D_j and D_i have the same order π, and D_j semantically implies D_i. The latter means that every assignment that satisfies D_j also satisfies D_i;

Reordering: D_i is an OBDD that is equivalent to an OBDDD_j with $j < i$ (note that D_i and D_j may have different orders).

We also say that $D_1, D_2, \dots, D_t$ is a refutation (or proof of unsatisfiability) of F if D_t is the constant false function.

We consider several different OBDD proof systems with different sets of allowed rules. When we need to denote some specific proof system, we indicate the rules specific for this system in brackets. For example, the OBDD($\wedge$) proof system uses only conjunction rule and hence, we may assume that all OBDDs in a proof have the same order. We use the notation π-OBDD($\wedge$) proof if all diagrams in this proof have order π.

It is possible to note that all the defined proof systems are Cook and Reckhow proof systems. Indeed, for OBDD($\wedge$) and OBDD($\wedge$, reordering) the following beforementioned properties of OBDDs imply this.

1. There is a polynomial-time algorithm that verifies whether a function represented by an OBDD is satisfiable.
2. Let D_1, D_2, D_3 be some OBDDs in the same order. It is possible to check whether $D_1 = D_2 \wedge D_3$ in polynomial time.
3. There is a polynomial-time algorithm that verifies whether an OBDD D_1 in an order π_1 and an OBDD D_2 in an order π_2 represent the same function.

To show that OBDD($\wedge$, weakening) and OBDD($\wedge$, weakening, reordering) are Cook and Reckhow proof systems we need to notice that D_j semantically implies D_i iff $D_j \wedge \neg D_i$ is unsatisfiable. However, using Properties 1 and 2, this can be checked in polynomial time.

It is important to notice that, Property 2 holds only for diagrams in the same order. Hence, the constraint that we can derive a conjunction of two OBDDs only

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

if they are in the same order is important if we want the proof systems to be Cook and Reckhow proof systems.

Remark 2.5. Note that if a formula ϕ has an OBDD refutation of size S, then for any partial substitution ρ to the variables of ϕ, there is an OBDD refutation (using the same set of rules) of $\phi|_\rho$ of size at most S.

Proof. Let $D_1, \dots, D_t$ be a refutation of ϕ of size S. We claim that $D_1|_\rho, \dots, D_t|_\rho$ is a refutation of ϕ of size at most S. First it is clear that $D_i|_\rho$ is also an OBDD and $\left|D_i|_\rho\right| \leq |D_i|$; therefore, $\sum_{i=1}^{t} |D_i| \leq \sum_{i=1}^{t} \left|D_i|_\rho\right|$. Let us now check that it is indeed a refutation of $\phi|_\rho$. First note that if D_i represents a clause of ϕ, then $D_i|_\rho$ represents a clause of $\phi|_\rho$. Secondly, it is clear that if D_i represents $D_j \wedge D_k$, then $D_i|_\rho$ represents $D_j|_\rho \wedge D_k|_\rho$. Finally, if D_i and D_j represent the same function, then $D_i|_\rho$ and $D_j|_\rho$ also represent the same function. ⊣

2.4. OBDD **algorithms for** SAT.

Aguirre and Vardi [1] proposed for the Boolean satisfiability problem the following family of algorithms based on OBDDs and the symbolic quantifier elimination.

The algorithm gets as an input a CNF formula F, it chooses some order π on the variables and creates both a π-ordered OBDD D (which initially is equal to the constant true function) and a set of clauses S (which initially consists of all clauses of the formula F). While S is not empty the algorithm applies one of the following three operations:

Join or conjunction: delete some clause C from S and replace D by a π-ordered BDD that represents the conjunction $D \wedge C$;

Projection: choose a variable x that has no occurrences in the clauses from S and replace D by a π-ordered BDD for the function $\exists x\ D$;

Reordering: choose a new order on variables π' and replace D by the equivalent π'-ordered diagram. Assign $\pi := \pi'$. (Note that Aguirre and Vardi did not consider this rule in [1]).

After every step of the algorithm, the following invariant is maintained: F is satisfiable if and only if $\bigwedge_{C \in S} C \wedge D$ is satisfiable. After the termination of the algorithm the set S is empty; if the diagram D has a path from the source to a vertex labeled by 1, then the algorithm returns “Satisfiable” and returns “Unsatisfiable” otherwise.

We refer to the algorithms of this type as $\mathrm{OBDD}(\wedge, \exists, \mathrm{reordering})$ algorithms. Besides, we use a similar notation for algorithms that use some of the rules: we just enumerate the used rules in the brackets. For example, the $\mathrm{OBDD}(\wedge)$ algorithms use only the conjunction rule and the $\mathrm{OBDD}(\wedge, \exists)$ algorithms use only the conjunction and projection rules.

Since join and projection for OBDDs may be performed in polynomial time and reordering may be performed in time polynomial on the sizes of the input and the output, the running time of an $\mathrm{OBDD}(\wedge, \exists, \mathrm{reordering})$ algorithm is polynomially related to the sum of the sizes of all states of the diagram D (here we ignore the time spent on choosing the next step of the algorithm).

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

2.5. Error-correcting codes. By a *code* we mean a subset of binary strings with a fixed length. A code $\mathcal{C} \subseteq \{0,1\}^n$ has a relative distance δ if for any two codewords $c_1, c_2 \in C$ the Hamming distance between c_1 and c_2 is at least δn.

A *linear code* is a set of all n-bits vectors $x = (x_1 \dots x_n)$ from some linear subspace in $\mathbb{F}_2^n$. If k is the dimension of this space, then the ratio k/n is called the *rate* of the code.

A linear code can be specified by a system of linear equations. For a code of dimension k this system should consist of $m \geq n-k$ linear equations involving n variables. The set of all solutions of the system should give exactly our code, so the rank of the system must be equal to $n-k$. If we require in addition that the equations in the system are linearly independent, then the number of equations is equal to $m = n-k$. The matrix of this linear system is called a *checksum matrix* of the code.

For a checksum matrix $(h_{i,j})$ over $\mathbb{F}_2$ we say that a column i *intersects* a row j, if $h_{j,i} = 1$. Further, we say that a tuple of columns $\langle i_1, \dots, i_s \rangle$ intersects some row j if at least one of the columns i_1, ..., i_s intersects row j.

We say that a code $\mathcal{C}$ recovers a ρ fraction of erasures by a list of size L (or $\mathcal{C}$ is (ρ, L)-erasure list-decodable) if for any $w \in \{0,1,?\}^n$ such that the number of ? in w does not exceed ρn, there exist at most L elements in $\mathcal{C}$ that are consistent with w. A string $s \in \{0,1\}^n$ is consistent with w if for all i, $w_i \in \{0,1\}$ implies $s_i = w_i$.

THEOREM 2.6 ([14, Lemma 2]). *If $\mathcal{C}$ is a code with relative distance δ, then for every $\varepsilon > 0$ the code C is $((2-\varepsilon)\delta, \frac{2}{\varepsilon})$-erasure list-decodable.*

We will use the following technical lemma later.

LEMMA 2.7. *Let A be an $m \times n$ checksum matrix of a linear (ρ, L)-erasure list decodable code. The matrix A' is the result of deleting r rows from A, where $r < m$. Then A' is a checksum matrix of a $(\rho, 2^r L)$-erasure list-decodable code.*

PROOF. Consider an arbitrary n-dimensional vector z with variables z_j at some ρn coordinates and fixed constants from $\{0,1\}$ at the other $(1-\rho)n$ coordinates. Then $A'z = 0$ can be viewed as a (nonuniform) system of linear equations for the ρn variables in z. We have to show that the number of solutions of this system is at most $2^r L$.

Let z' be the n-dimensional vector that contains variables at the same positions as the initial vector z, and with zeros at all other positions. Since A is the checksum matrix of a (ρ, L)-erasure list decodable code, the system $Az' = 0$ has at most L solutions. Since the system $Az' = 0$ is homogeneous, we conclude that the number of its solutions is exactly $2^{\rho n - \ell}$, where ℓ is the rank of this linear system.

Note that ℓ can be less than the rank of the matrix A, since not all n components of z' contain variables (some of them are set to the constant value 0). However, the space of solutions of the system $Az' = 0$ must have the same dimension as the space of solutions of the system $Az = 0$ (these systems differ only in the fixed constant terms). Further, when we delete from the linear system $Az = 0$ some r equations (the r rows that belong to A but not to A'), we increase the dimension of the space of solutions by at most r. Therefore, the system $A'z = 0$ has at most $2^r L$ solutions, and we are done. ⊣

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

§3. Lower bounds for OBDD($\wedge$, reordering).

3.1. Tseitin formulas. In this section we prove an exponential lower bound on the size of OBDD($\wedge$, reordering) proofs of Tseitin formulas.

The canonical CNF representation of a predicate $P(x_1, x_2, \dots, x_n)$ is the conjunction of clauses $(x_1^{a_1} \vee x_2^{a_2} \vee \dots \vee x_2^{a_n})$ over all $a_1, a_2, \dots, a_n \in \{0,1\}$ such that $P(a_1, a_2, \dots, a_k) = 0$, where x^0 denotes x and x^1 denotes $\neg x$. Notice that the canonical representation of constant true predicate is the empty CNF.

A Tseitin formula $\mathtt{TS}_{G,c}$ is a CNF formula based on an undirected graph $G(V, E)$ and a labelling function $c : V \to \{0,1\}$. Every edge $e \in E$ has the corresponding propositional variable x_e. If G has an isolated vertex v with $c(v) = 1$, then $\mathtt{TS}_{G,c}$ is constant false and we represent it by the empty clause, otherwise $\mathtt{TS}_{G,c}$ is the conjunction of *the parity conditions* for all vertices v. If $e_1, e_2, \dots, e_k$ are all edges incident to v, then the parity condition of v is the canonical CNF representation of the predicate $x_{e_1} + x_{e_2} + \dots + x_{e_k} \equiv c(v) \pmod 2$. (Notice that if v is isolated, then $c(v) = 0$, hence the canonical representation of the parity condition of v is the empty CNF). Notice that if all degrees of G are at most d, then $\mathtt{TS}_{G,c}$ contains at most $2^{d-1}|V|$ clauses.

If $\sum\limits_{v \in U} c(v) \equiv 1 \pmod 2$ for some connected component $U \subseteq V$, then the Tseitin formula is unsatisfiable. Indeed, if we sum up (modulo 2) all equalities stated in vertices from U we get $0 = 1$ since every variable has exactly 2 occurrences. If $\sum\limits_{v \in U} c(v) \equiv 0 \pmod 2$ for every connected component U, then the Tseitin formula is satisfiable ([28, Lemma 4.1]).

Let φ be a CNF formula and ρ be a partial assignment of its variables. We say that $\phi|_\rho$ is the empty clause, if ρ falsifies a clause of C and $\phi|_\rho$ is the conjunction of all $C|_\rho$ over clauses C of ϕ that is not satisfied by ρ, where $C|_\rho$ is a clause that is obtained from C by the deletion of all literals with variables from the support of ρ.

Let $\mathtt{TS}_{G,c}$ be a Tseitin formula and ρ be a partial substitution of its variables. Let $E' \subseteq E$ be set of edges corresponding to the support of ρ. We say that a labeling function $c' : V \to \{0,1\}$ corresponds to ρ if for every vertex v, $c'(v) = \sum_e \rho(x_e) \bmod 2$, where e goes over all edges from E' that are incident to v. The following proposition is straightforward:

Proposition 3.1. *Let $\mathtt{TS}_{G,c}$ be a Tseitin formula, ρ be a partial substitution of its variables and c' be a labeling function corresponding to ρ. Then $\mathtt{TS}_{G,c}|_\rho = \mathtt{TS}_{G'',c+c'}$, where G'' is a graph with vertices V and edges $E \setminus E'$, and $c + c'$ is the sum of the functions c and c' modulo 2.*

Note that if formulas $\mathtt{TS}_{G,c}$ and $\mathtt{TS}_{G,c'}$ are satisfiable and $c \neq c'$, then $\mathtt{TS}_{G,c}$ and $\mathtt{TS}_{G,c'}$ are different functions; moreover, any satisfying assignment of $\mathtt{TS}_{G,c}$ does not satisfy $\mathtt{TS}_{G,c'}$.

In the following, we denote the number of connected components of a graph G by $\sharp G$.

Theorem 3.2. *Let graph G with vertices V and edges E have the following property: for some $m \in [|E|]$ and $k > 0$ for all subsets $E' \subseteq E$ of size m the inequality $\sharp G' + \sharp G'' \leq k$ holds, where G' and G'' are graphs with vertices V and edges E' and $E \setminus E'$ respectively. If $\mathtt{TS}_{G,c}$ is satisfiable and $\Pi = (\Pi_0, \Pi_1)$ be a partition of the variables of $\mathtt{TS}_{G,c}$ such that*

$|\Pi_0| = m$, *then the communication complexity of* $\mathtt{TS}_{G,c}$ *with respect to* Π *is at least* $|V| - k + 1$.

Proof. Let E' be the set of the edges corresponding to the variables from Π_0. We show that there are at least $2^{|V|-k}$ substitutions to the variables from Π_0 such that applying each of these substitutions to $\mathtt{TS}_{G,c}$ results in $2^{|V|-k}$ satisfiable functions with disjoint sets of satisfying assignments. Then due to Lemma 2.3, the communication complexity of $\mathtt{TS}_{G,c}$ is at least $|V| - k + 1$.

By Proposition 3.1 it is sufficient to prove a lower bound for the number of different c' that corresponds to a partial substitution with the support Π_0 such that $\mathtt{TS}_{G'',c+c'}$ is satisfiable. The required properties of c' can be described by a system of linear equations with variables $c'(v)$ for $v \in V$: for every connected component U of graph G' with vertices V and edges E' we put down the equation: $\sum\limits_{v\in U} c'(v) = 0$ (this subsystem states that c' corresponds to a substitution or in other words that $\mathtt{TS}_{G',c'}$ is satisfiable) and for each connected component W of G'' we put down the equation: $\sum_{v\in W} c(v) + c'(v) = 0$ (this subsystem corresponds to the satisfiability of $\mathtt{TS}_{G'',c+c'}$).

The system has a solution since $\mathtt{TS}_{G,c}$ is satisfiable. There are $|V|$ variables and at most $\sharp G' + \sharp G'' \le k$ equations. Hence, there is at least $2^{|V|-k}$ solutions. Different solutions correspond to different satisfiable formulas $\mathtt{TS}_{G'',c+c'}$ and thus their sets of satisfying assignments are disjoint. ⊣

We will apply Theorem 3.2 to algebraic expanders.

Definition 3.3. A graph G with vertices V and edges E is an (n,d,α)-algebraic expander, if $|V| = n$, the degree of any vertex in V equals d and the absolute value of the second largest eigenvalue of the adjacency matrix of G is not greater than αd.

It is well known that for all $\alpha > 0$ and all large enough constants d there exists a family G_n of (n,d,α)-algebraic expanders. There are explicit constructions such that G_n can be constructed in poly(n) time [21]. Also, it is known that a random d-regular graph is a good expander with high probability.

Lemma 3.4 ([2]). *(Expander mixing lemma) Let* $G(V,E)$ *be an* (n,d,α)*-expander. For any two subsets* $S,T \subseteq V$ *the following inequality holds:* $\left||E(S,T)| - \frac{d|S||T|}{n}\right| \le \alpha d\sqrt{|S||T|}$, *where* $E(S,T) = \{(u,v) \mid u \in S, v \in T, (u,v) \in E\}$.

Lemma 3.5 ([6]). *(Cheeger's inequality) Let* $G(V,E)$ *be an* (n,d,α)*-expander. For every set* $S \subseteq V$, $|S| \le \frac{n}{2}$ *the following inequality holds:* $|E(S, V\setminus S)| \ge \frac{1-\alpha}{2} d|S|$.

Theorem 3.6. *Let* $G(V,E)$ *be an* (n,d,α)*-algebraic expander for* $\alpha < \frac{1}{2}$. *For any* $E' \subseteq E$, *if* $|E'| = \frac{n}{4d}$, *then* $\sharp G' + \sharp G'' \le n(1-\varepsilon) + 2$, *where* G' *is a graph with vertices* V *and edges* E', G'' *is a graph with vertices* V *and edges* $E \setminus E'$ *and* $\varepsilon = \frac{1}{8d(\alpha d+1)} - \frac{1}{d^2}$. *Note that* $\varepsilon > 0$ *if* $\alpha < 1/32$ *and* $d \ge 4$.

Proof. At first we prove the following proposition.

Proposition 3.7. *For any connected component* C *of* G'' *either* $|C| < \frac{4|E'|}{d}$ *or* $|C| > \frac{|V|}{2}$. ⊣

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Proof. Assume for the sake of contradiction that there is a connected component C such that $\frac{4|E'|}{d} < |C| \le \frac{|V|}{2}$. By Lemma 3.5 applied to the graph G, we have that $|E(C, V\setminus C)| \ge \frac{1-\alpha}{2}d|C| > \frac{d}{4}|C| \ge |E'|$. Hence, there are edges between C and $V\setminus C$ in G'', we get a contradiction. ⊣

Let us prove that $\sharp G'' \le \frac{4|E'|}{d} + 2$. By Proposition 3.7, if the size of a connected component of G'' is at least $\frac{4|E'|}{d}$, then the size of this component is greater than $\frac{|V|}{2}$; hence, there is at most one component with size at least $\frac{4|E'|}{d}$. Consider all other components. If the number of these components is at least $\frac{4|E'|}{d}+1$, we choose several of them such that the total number of vertices in the chosen components is in the interval $\left[\frac{4|E'|}{d}+1, \frac{8|E'|}{d}+1\right]$. We denote the set of vertices in these components by T. Since T consists of several connected components in G'' we have that $|E(T, V\setminus T)| = 0$ in G'' and, thus, $E(T, V\setminus T) \le |E'|$ in G. Since $|T| \le \frac{8|E'|}{d}+1 = \frac{2n}{d^2}+1 \le \frac{n}{2}$, it follows from Lemma 3.5 that $|E(T, V\setminus T)| \ge \frac{1-\alpha}{2}d|T| > |E'|$ in G; so we get a contradiction.

Now we prove that $\sharp G' \le n - \frac{|E'|}{2(\alpha d+1)}$. Let S be the set of vertices incident to E'. Note that if H is the graph with vertices S and edges E' then $|V| - \sharp G' = |S| - \sharp H$ and the degree of each vertex of H is at least one. Hence, $\sharp H \le \frac{|S|}{2}$ and $\sharp G' \le n - \frac{|S|}{2}$.

By Lemma 3.4 for the graph G we have that $|E(S,S)| \le \frac{d}{|V|}|S|^2 + \alpha d|S| \le |S|(\alpha d + \frac{d}{|V|}|S|) \le |S|(\alpha d + \frac{2d}{|V|}|E'|) \le |S|(\alpha d+1)$. However $|E(S,S)| \ge |E'|$ since we count each edge from E' at least once in $E(S,S)$. Hence, $|S| \ge \frac{|E'|}{\alpha d+1}$ and $\sharp G' \le n - \frac{|E'|}{2(\alpha d+1)}$.

Finally, we get $\sharp G' + \sharp G'' \le n - \frac{|E'|}{2(\alpha d+1)} + \frac{4|E'|}{d} + 2 = n(1 - (\frac{1}{8d(\alpha d+1)} - \frac{1}{d^2})) + 2$.

Corollary 3.8. *Let G be an (n,d,α)-algebraic expander with $\alpha < \frac{1}{32}$ and $d \ge 4$. Assume that a Tseitin formula* $\mathtt{TS}_{G,c}$ *is satisfiable. Then for every partition Π of the variables of* $\mathtt{TS}_{G,c}$ *such that $|\Pi_0| = \frac{n}{4d}$, communication complexity of* $\mathtt{TS}_{G,c}$ *is at least $\Omega(n)$.*

Proof. Follows from Theorems 3.2 and 3.6. ⊣

Corollary 3.9. *If a graph G is an (n,d,α)-algebraic expander for $\alpha < \frac{1}{32}$ and $d \ge 4$, and a Tseitin formula* $\mathtt{TS}_{G,c}$ *is satisfiable, then every* OBDD *representing* $\mathtt{TS}_{G,c}$ *has size at least $2^{\Omega(n)}$.*

Proof. Follows from Corollary 3.8 and Lemma 2.1. ⊣

Corollary 3.10. *Let graph H_n be obtained from an (n,d,α)-expander $G(V,E)$ with $\alpha < \frac{1}{32}$ and $d \ge 4$ by removing $o(n)$ edges. If* $\mathtt{TS}_{H_n,c}$ *is satisfiable, then every* OBDD *representation of* $\mathtt{TS}_{H_n,c}$ *has size at least $2^{\Omega(n)}$.*

Proof. Let M be the set of edges of H_n, $E' \subset M$ such that $|E'| = \frac{n}{4d}$, and H', H'', G'' be graphs with vertices V and edges E', $M\setminus E'$ and $E\setminus E'$ respectively. Obviously $\sharp H'' \le \sharp G'' + o(n)$. By Corollary 3.6, $\sharp H' + \sharp H'' \le (1-\varepsilon)n + 2$. Hence, $\sharp H' + \sharp G'' \le (1-\varepsilon)n + 2 + o(n)$. Therefore, by Theorem 3.2 and Lemma 2.1, size of an OBDD for $\mathtt{TS}_{H_n,c}$ is at least $2^{\Omega(n)}$. ⊣

LEMMA 3.11 [29]. 1. *If for functions $f_1 : \{0,1\}^n \to \{0,1\}$ and $f_2 : \{0,1\}^n \to \{0,1\}$ there exist π-ordered BDDs (for some order π) with sizes k_1 and k_2 respectively then there exists a π-ordered BDD of size at most k_1k_2 for $f_1 \wedge f_2$.*

2. *If for a function $f : \{0,1\}^n \to \{0,1\}$ there exists a π-ordered BDD of a size k then for any substitution ρ there exists a π-ordered BDD for $f|_\rho$ of size at most k.*

LEMMA 3.12. *Assume that all* OBDD*s representing a CNF formula ϕ have the size at least k. Assume that a CNF formula ψ can be obtained from ϕ by removing several clauses that are dependent on at most t variables in total. Then the size of any* OBDD *representing ψ is at least $k2^{-t-1}$.*

PROOF. The conjunction of the removed clauses is a function of t variables, hence it can be represented as OBDD of size at most 2^{t+1} for every order. Thus if ψ has OBDD representation of size less than $k2^{-t-1}$ by the first part of Lemma 3.11, then ϕ has OBDD representation of size less than k. ⊣

LEMMA 3.13. *Let $G(V,E)$ be an (n,d,α)-algebraic expander with $\alpha < \frac{1}{20}$ and $d \geq 50$. Then for n large enough G is connected and it remains connected after removing of any vertex or any two vertices and edges of the shortest path between them.*

PROOF. It is sufficient to prove that G is connected after removing any two vertices and edges of the shortest path between them. Consider arbitrary two vertices $x, y \in V$ and consider an arbitrary set $S \subseteq V \setminus \{x,y\}$ such that $|S| \leq |V \setminus (\{x,y\} \cup S)|$. We will show that there is $v \in S$ such that $|E(\{v\}, V \setminus (\{x,y\} \cup S))| \geq 3$. This will imply that after removing x and y and edges of the shortest xy-path from G the resulting graph is connected. Indeed, if it is not connected, then we get a contradiction for S that is equal to the smallest connected component. In this case the shortest xy-path may contain at most 2 edges incident to v, hence, there is an edge that goes from S to $V \setminus (\{x,y\} \cup S)$.

By Cheeger's inequality (Lemma 3.5), $|E(S, V \setminus S)| \geq \frac{1-\alpha}{2}d|S|$. Since the degrees of x and y are exactly d, we get $|E(S, V \setminus (S \cup \{x,y\}))| \geq \frac{1-\alpha}{2}d|S| - 2d$. If $|E(S, V \setminus (S \cup \{x,y\}))| > 2|S|$, then the required v exists by the averaging principle. Assume that $|E(S, V \setminus (S \cup \{x,y\}))| \leq 2|S|$, hence $\frac{1-\alpha}{2}d|S| - 2d \leq 2|S|$ and $(\frac{1-\alpha}{2}d - 2)|S| \leq 2d$, and since $\alpha < \frac{1}{20}$ and $d \geq 50$, then $|S| < 5$.

If $|S| < 5$, then the expander mixing lemma implies that $|E(S, \{x,y\})| \leq \frac{d}{n}2|S| + \alpha d\sqrt{2|S|} \leq 5\alpha d$ for n large enough. By Cheeger's inequality $|E(S, V \setminus S)| \geq \frac{1-\alpha}{2}d|S|$ and $|E(S, V \setminus (S \cup \{x,y\}))| \geq \frac{1-\alpha}{2}d|S| - 5\alpha d \geq 10 > 2|S|$ for $\alpha < \frac{1}{20}$ and $d > 50$; and this is a contradiction. ⊣

LEMMA 3.14. *Let G be an (n,d,α)-algebraic expander with $d \geq 4$, $\alpha < \frac{1}{32}$. Let Tseitin formula $\mathrm{TS}_{G,c}$ be unsatisfiable and Φ denote the formula that can be obtained from $\mathrm{TS}_{G,c}$ by removing one arbitrary clause. Then Φ is satisfiable and every* OBDD *representation of Φ has the size $2^{\Omega(n)}$.*

PROOF. Assume that $\mathrm{TS}_{G,c} = \Phi \wedge C$, where C is a clause that corresponds to the parity condition of a vertex v. Let H be the result of removing the vertex v from G. By Lemma 3.13 the graph H is connected. Let us make a substitution to all variables of the clause C that falsifies C. Let Φ' denote the result of this substitution applied to Φ. Since the substitution falsifies the parity condition of the vertex v, Φ'

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

corresponds to a satisfiable Tseitin formula based on H. By Corollary 3.10, the size of every OBDD for Φ' is at least $2^{\Omega(n)}$. (Formally we apply Corollary 3.10 to the graph H with one additional isolated vertex v. Note that a Tseitin formula does not change if we add an isolated vertex to the graph). Φ' is the result of the substitution applied to Φ, hence, the size of every OBDD representing Φ is at least $2^{\Omega(n)}$. ⊣

Now we are ready to prove that Tseitin formulas based on expanders are hard for OBDD($\wedge$, reordering).

Theorem 3.15. *Let G be an (n, d, α)-algebraic expander with $d \geq 50$ and $\alpha < \frac{1}{32}$. Then any* OBDD($\wedge$, reordering) *refutation of any unsatisfiable Tseitin formula* $\mathtt{TS}_{G,c}$ *has size at least* $2^{\Omega(n)}$.

Before we present the formal proof of the theorem let us sketch the argument. We consider the last step of the OBDD proof: the conjunction of OBDDs F_1 and F_2 is the identically false function but both F_1 and F_2 are satisfiable. Both F_1 and F_2 are conjunctions of several clauses of $\mathtt{TS}_{G,c}$. We use the fact that a satisfiable Tseitin formula based on an expander has only exponential sized OBDDs. Moreover, by Corollary 3.10, if the underlying graph differs from some expander by $o(n)$ edges, then any of its OBDD representations has also an exponential size.

Note that F_1 and F_2 together contain all clauses of $\mathtt{TS}_{G,c}$. Assume that there are two nonadjacent vertices u and v such that F_1 does not contain a clause C_u from the parity condition of u and F_2 does not contain a clause C_v from the parity condition of v. We consider two partial substitutions ρ_1 and ρ_2 that are both defined on the edges adjacent to u and v and on the edges of a fixed shortest path p between u and v. The substitutions ρ_1 and ρ_2 assign opposite values to edges of the path p and are consistent on all other edges. The substitution ρ_1 satisfies C_v and refutes C_u and ρ_2 satisfies C_u and refutes C_v.

By the construction $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is a satisfiable Tseitin formula based on the graph that is obtained from G by deletion of the vertices u and v and all edges from the path p (to be honest, it is not always true and it is also possible that $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ does not contain some clauses from the parity conditions of vertices from p). The size of an OBDD representation of such a formula is exponential since the underlying graph is obtained from an expander by removing at most $o(n)$ vertices. (Note that p is the shortest path in the expander, thus p contains at most $O(\log n)$ edges.) Hence, we get that either F_1 or F_2 has an exponential size in the given order.

Proof. We consider the last step of the proof: the conjunction of OBDDs F_1 and F_2 is the identically false function but F_1 and F_2 are satisfiable. Both F_1 and F_2 are conjunctions of several clauses of $\mathtt{TS}_{G,c}$. Every clause of $\mathtt{TS}_{G,c}$ is either in F_1 or in F_2 since otherwise $F_1 \wedge F_2$ is satisfiable by Lemma 3.14. Our goal is to prove that either F_1 or F_2 has size $2^{\Omega(n)}$.

We consider two cases:

1. There exist two nonadjacent vertices u and v from G such that F_1 does not include some clause C_v from the parity condition of v and F_2 does not include some clause C_u from the parity condition of u.

   Consider a shortest path p from v to u. Let e_v be the first edge of p and e_u be the last edge ($e_v \neq e_u$ since u and v are nonadjacent). Consider two substitutions ρ_1 and ρ_2 with the same support: all edges that are incident to u or v and all

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

edges from p. Substitutions ρ_1 and ρ_2 are consistent on edges that are out of p: all edges that are adjacent to u or v but not in p have values that do not satisfy C_u and C_v (this is possible since u and v are nonadjacent). ρ_1 substitute zeros to all edges from p except e_u and e_v and substitute a value to e_v that does not satisfy C_v and a value to e_u that satisfies C_u. ρ_2 substitute ones to all edges from p except e_u and e_v and substitute a value to e_v that satisfies C_v and a value to e_u that does not satisfy C_u. So edges from p have opposite values in ρ_1 and ρ_2. We claim that ρ_1 satisfies the parity condition of u and all clauses from the parity condition of v except C_v. Indeed, ρ_1 refutes C_v, since the parity condition is represented in the canonical form, it satisfies all other clauses from the parity condition of v. By the construction, ρ_1 satisfies only one literal of C_u, hence ρ_1 satisfies the parity condition of vertex u (it is easy to see that any assignment falsifying the parity condition of u satisfies even number of literals of C_u). Similarly, we get that ρ_2 satisfies all clauses from the parity condition of u except C_u and satisfies the parity condition of v. Since F_1 does not contain C_v and F_2 does not contain C_u, ρ_1 satisfies all clauses from the parity conditions of u and v that are in F_1 and ρ_2 satisfies all clauses from the parity conditions of u and v that are in F_2.

Consider the graph G' that is obtained from G by removing u, v and all edges from the path p. The graph G' is connected since by Lemma 3.13 after removing any two vertices with a shortest path between them the graph G remains connected (we may assume that n is large enough).

Let $c_i : V \to \{0,1\}$ for $i \in \{1,2\}$ be a labeling function corresponding to ρ_i. By the construction of ρ_1 and ρ_2, c_1 and c_2 differs only in u in v and $c_1(u) = c(u), c_1(v) = c(v)+1$. Let $c' = c + c_1$ and c'' be the restriction of c' on $V \setminus \{u,v\}$. Notice that by Proposition 3.1, $\mathtt{TS}_{G,c}|_{\rho_1} = \mathtt{TS}_{G',c''}$. We claim that $\mathtt{TS}_{G',c''}$ is satisfiable. Since G'' is connected, we have to verify that $\sum_{w \in V\setminus\{u,v\}} c''(w)$ is even. Indeed,

$$\sum_{w\in V\setminus\{u,v\}} c''(w) = \sum_{w\in V\setminus\{u,v\}} (c(w)+c_1(w)) = 1 + \sum_{w\in V}(c(v)+c_1(v)) \overset{(\mathtt{TS}_{G,c} \text{ is unsatisfiable})}{=} \sum_{w\in V} c_1(w) \overset{(c_1 \text{ corresponds } \rho_1)}{=} 0.$$

We consider the conjunction $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$. We claim that $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ can be obtained from $\mathtt{TS}_{G',c''}$ by removing of $O(\log n)$ clauses corresponding to parity conditions of vertices from the path p. Let us describe clauses of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$; all of them comes from the parity conditions of vertices in $\mathtt{TS}_{G,c}$. Assume that $F_1|_{\rho_1} \wedge F_2|_{\rho_2} = \bigwedge_{w\in V} P_w$, where P_w is the conjunction of clauses that are obtained from the parity condition of w. By the construction, none of clauses comes from the parity conditions of vertices u and v, hence P_u and P_v are the empty conjunctions. For every vertex w that is not in p, the union of clauses of F_1 and F_2 contains all clauses from the parity condition of w, substitutions ρ_1 and ρ_2 are consistent for all edges that are incident w, hence, P_w is exactly the parity condition of w in the formula $\mathtt{TS}_{G',c''}$. Consider a vertex w from p such that $w \notin \{u,v\}$. As mentioned above $c_1(w) = c_2(w)$, hence if we apply ρ_1 to the parity condition of w we will get the same set of clauses if we apply

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

ρ_2 to the parity condition of w, but orders of clauses will be different. Hence, every clause of P_w is a clause from the parity condition of w in $\mathtt{TS}_{G',c''}$ but it is not necessary that P_v contains all clauses from the parity condition of w in $\mathtt{TS}_{G',c''}$. Since G is an (n,d,α)-expander, its diameter is $O(\log n)$, hence $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ can be obtained from $\mathtt{TS}_{G',c''}$ by removing of $O(\log n)$ clauses.

The size of any OBDD representation of the formula $\mathtt{TS}_{G',c''}$ is at least $2^{\Omega(n)}$ by Corollary 3.10, since G' can be obtained from an (n,d,α)-algebraic expander by the removing of $O(\log n)$ edges (formally we add two isolated vertices u and v to G' with zero labeling function on them; adding isolated vertices does not change Tseitin formulas). Then by Lemma 3.12, any OBDD representing $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has the size at least $2^{\Omega(n)}$. Thus, by Lemma 3.11, for every given order of variables π either $F_1|_{\rho_1}$ or $F_2|_{\rho_2}$ has the size of the minimal π-OBDD at least $2^{\Omega(n)}$. Hence, the minimal π-OBDD for F_1 or F_2 has the size of the at least $2^{\Omega(n)}$.

2. In the second case there are no such nonadjacent vertices. Since F_1 is not identically false, there exists a vertex u such that F_1 does not include a clause that corresponds to a vertex u and by the assumption F_2 does not include clauses only for the vertex u and its neighbors. In this case F_2 differs from a Tseitin formula without one clause by a constant number of clauses that depend on a constant number of variables. Thus any OBDD representation of F_2 by Lemmas 3.14 and 3.12 has size at least $2^{\Omega(n)}$. ⊣

3.2. Pigeonhole principle. In this section we consider formulas that encode the pigeonhole principle and prove an exponential lower bound on the size of OBDD($\wedge$, reordering) proofs of them.

Let m and n be integers and $p_{i,j}$ for $i \in [m]$ and $j \in [n]$ be different variables; $p_{i,j}$ states whether the ith pigeon is in the jth hole or not. The formula $\mathtt{PHP}^m_n$ has two types of clauses. The clauses of the first type (long clauses) state that every pigeon is in at least one hole: $\bigvee\limits_{j=1}^{n} p_{i,j}$ for all $i \in [m]$. The clauses of the second type (short clauses) state that in every hole there is at most one pigeon: $\neg p_{i,k} \vee \neg p_{j,k}$ for all $k \in [n]$ and all $i \neq j \in [m]$.

Lemma 3.16. *For every partition Π such that $|\Pi_0| = \frac{n^2}{96}$, the communication complexity of $\mathtt{PHP}^n_n$ with respect to Π is at least $\Omega(n)$.*

Proof. Let us fix some partition of variables Π such that $|\Pi_0| = \frac{n^2}{96}$.

Consider a bipartite graph $G(L,R,E)$ with two parts L and R associated with $[n]$ and the set of edges E associated with Π_0, i.e., $(i,j) \in E$ iff $p_{i,j} \in \Pi_0$. Let $G'(L,R,E')$ be the complement bipartite graph, i.e., $E' = [n] \times [n] \setminus E$. Let v_1, v_2, ..., v_{2n} be the vertices of $R \cup L$ sorted by decreasing degrees in G': $\deg v_1 \geq \deg v_2 \geq \cdots \geq \deg v_{2n}$.

Let k be the maximum number such that $\deg v_k \geq 3k$. Note that $\frac{n^2}{48} = 2|E'| \geq \sum\limits_{i=1}^{k} \deg v_i \geq 3k^2$, and therefore $k \leq \frac{n}{12}$. Further, the definition of k implies that if $k < 2n$, then $\deg v_{k+1} < 3(k+1)$; i.e., $3k+2 \geq \deg v_{k+1} \geq \deg v_{k+2} \geq \cdots \geq \deg v_{2n}$.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Hence, $\frac{n^2}{48} = 2|E'| = \sum_{i=1}^{2n} \deg v_i \leq nk + (2n-k)(3k+2) = nk + 6nk + 4n - 3k^2 - 2k \leq 11nk$. Therefore, $k \geq \frac{n}{11 \cdot 48}$. Let $B = \{v_1, v_2, \dots, v_k\}$. Note that every vertex from B is connected in G' with at least $2k$ vertices outside of B.

Let $\mathfrak{M}$ be the set of all matchings in G' of size k that cover B and do not connect two vertices from B. Let us prove that $|\mathfrak{M}| \geq \frac{(2k)!}{k!}$. Note that the number of possibilities for the ith vertex from B to find a pair is at least $3k - |B| - (i-1) = 2k - i + 1$. Hence, there are at least $\frac{(2k)!}{k!}$ matchings of this type. So there are at least $\frac{(2k)!}{k!k!} = \binom{2k}{k}$ different sets that are sets of endpoints of matchings from $\mathfrak{M}$.

For every $M \in \mathfrak{M}$ we define a partial substitution ρ_M with the support Π_0 such that $\rho_M(p_{i,j}) = 1$ if $(i,j) \in M$ and $\rho_M(p_{i,j}) = 0$ if $(i,j) \in E \setminus M$. We claim that $\mathtt{PHP}^n_n|_{\rho_M}$ is satisfiable for all $M \in \mathfrak{M}$.

Let $V(M)$ denote the set of endpoints of edges from M. $\mathtt{PHP}^n_n|_{\rho_M}$ is satisfiable iff the graph $G - V(M)$ (which is obtained from G by removing all vertices from M) has a perfect matching. Since M is a matching, $G - V(M)$ has the same number of vertices in its parts. We are going to use Hall's theorem to show that $G - V(M)$ has a perfect matching. To this end we have to show that every set $A \subseteq L \setminus V(M)$ has at least $|A|$ neighbors in $R \setminus V(M)$. Consider two cases.

- In the first one, $|A| \leq \frac{n}{2}$. Note that every vertex from A has degree at least $n - 2k \geq \frac{n}{2}$ in $G - V(A)$. Therefore, it is clear that there are at least $|A|$ neighbors of A in $R \setminus V(M)$.
- In the second case $|A| > \frac{n}{2}$. We show that every vertex from $R \setminus V(M)$ has a neighbor in A and thus the number of neighbors of A is precisely $|R \setminus V(M)| = |L \setminus V(M)| \geq |A|$. Indeed, every vertex from $R \setminus V(M)$ has degree at least $n - 2k \geq \frac{n}{2}$ in $G - V(M)$ and $|L \setminus A| < \frac{n}{2}$, so it has a neighbor in A.

Let us consider $M_1, M_2 \in \mathfrak{M}$ with $V(M_1) \neq V(M_2)$. Functions $\mathtt{PHP}^n_n|_{\rho_{M_1}}$ and $\mathtt{PHP}^n_n|_{\rho_{M_2}}$ have disjoint sets of satisfiable assignments since M_1 and M_2 are matchings covering different sets of vertices. Thus, by Lemma 2.3, the communication complexity of $\mathtt{PHP}^n_n$ is at least $\left\lfloor \log \binom{2k}{k} \right\rfloor + 1 = \Omega(n)$. ⊣

Corollary 3.17. *Every* OBDD *representation of* $\mathtt{PHP}^n_n$ *has size at least* $2^{\Omega(n)}$.

Theorem 3.18. *Any* OBDD($\wedge$, reordering) *refutation of the pigeonhole principle formula* $\mathtt{PHP}^{n+1}_n$ *has size at least* $2^{\Omega(n)}$.

Proof. We consider the last step of the OBDD proof: the conjunction of OBDDs F_1 and F_2 is the identically false function but F_1 and F_2 are satisfiable. Both F_1 and F_2 represent conjunctions of several clauses of $\mathtt{PHP}^{n+1}_n$ and have the same order π. Every clause of $\mathtt{PHP}^{n+1}_n$ is either in F_1 or in F_2 since otherwise $F_1 \wedge F_2$ is satisfiable. Our goal is to prove that either F_1 or F_2 has size $2^{\Omega(n)}$.

We consider three cases and apply Corollary 3.17 in each of them:

1. Assume that there exist an $i_1 \in [n+1]$ and an $i_2 \in [n+1]$ $(i_1 \neq i_2)$ such that F_1 does not contain a long clause that corresponds to i_1 and F_2 does not contain a long clause that corresponds to i_2. Let us fix any $j \in [n]$ and consider substitutions ρ_1 and ρ_2 with the support

$$\{p_{s,t} | (s,t) \in \{i_1, i_2\} \times [n] \cup [n+1] \times \{j\}\}$$

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

that substitute zero to all variables with two exceptions: $\rho_1(p_{i_2,j}) = 1$ and $\rho_2(p_{i_1,j}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-1}_{n-1}$ and thus, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is of size at least $2^{\Omega(n)}$. Hence, either F_1 or F_2 has size of its minimal π-OBDD at least $2^{\Omega(n)}$.

2. Assume that there are $i_1, i_2 \in [n+1], j \in [n]$ and $i \in [n+1]$ such that F_1 does not contain the short clause $(\neg p_{i_1,j} \vee \neg p_{i_2,j})$ and F_2 does not contain a long clause that corresponds to i.

   Let us consider two sub-cases.

   - In the first case $i \notin \{i_1, i_2\}$. Let us fix any $j' \in [n] \setminus \{j\}$ and consider substitutions ρ_1 and ρ_2 with the support

     $$\{p_{s,t} \mid (s,t) \in \{i_1, i_2, i\} \times [n] \cup [n+1] \times \{j, j'\}\}$$

     that substitute zero to all variables with five exceptions: $\rho_1(p_{i_1,j}) = 1$, $\rho_1(p_{i_2,j}) = 1$, $\rho_1(p_{i,j'}) = 1$, $\rho_2(p_{i_1,j}) = 1$, and $\rho_2(p_{i_2,j'}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-2}_{n-2}$; hence, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$. Hence, either for F_1 or for F_2, the size of its minimal π-OBDD at least $2^{\Omega(n)}$.
   - In the second case $i \in \{i_1, i_2\}$. Without loss of generality $i = i_1$. Consider substitutions ρ_1 and ρ_2 with the support

     $$\{p_{s,t} \mid (s,t) \in \{i_1, i_2\} \times [n] \cup [n+1] \times \{j\}\}$$

     that substitute zero to all variables with three exceptions: $\rho_1(p_{i_1,j}) = 1$, $\rho_1(p_{i_2,j}) = 1$, and $\rho_2(p_{i_1,j}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-1}_{n-1}$; hence, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$. Hence, either for F_1 or for F_2, the size of the minimal π-OBDD at least $2^{\Omega(n)}$.

3. Assume that there are $i_{1,1}, i_{1,2} \in [n+1]$, $j_1 \in [n]$, $i_{2,1}, i_{2,2} \in [n+1]$ and $j_2 \in [n]$ such that F_1 does not contain the short clause $(\neg p_{i_{1,1},j_1} \vee \neg p_{i_{1,2},j_1})$ and F_2 does not contain the short clause $(\neg p_{i_{2,1},j_2} \vee \neg p_{i_{2,2},j_2})$. Let us consider five cases.

   - The first case is when $i_{1,1}, i_{1,2} \notin \{i_{2,1}, i_{2,2}\}$ and $j_1 \neq j_2$. Let us fix any $j \in [n] \setminus \{j_1, j_2\}$ and consider substitutions ρ_1 and ρ_2 with the support

     $$\{p_{s,t} \mid (s,t) \in \{i_{1,1}, i_{1,2}, i_{2,1}, i_{2,2}\} \times [n] \cup [n+1] \times \{j_1, j_2, j\}\}$$

     that substitute zero to all variables with eight exceptions: $\rho_1(p_{i_{1,1},j_1}) = 1$, $\rho_1(p_{i_{1,2},j_1}) = 1$, $\rho_1(p_{i_{2,1},j_2}) = 1$, $\rho_1(p_{i_{2,2},j}) = 1$, $\rho_2(p_{i_{2,1},j_2}) = 1$, $\rho_2(p_{i_{2,2},j_2}) = 1$, $\rho_2(p_{i_{1,1},j_1}) = 1$, and $\rho_2(p_{i_{1,2},j}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-3}_{n-3}$ and thus, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$.
   - The second case is when $i_{1,1} \notin \{i_{2,1},\, i_{2,2}\}$, $i_{1,2} \in \{i_{2,1},\, i_{2,2}\}$, and $j_1 \neq j_2$. Without loss of generality $i_{1,2} = i_{2,1}$. Let us consider substitutions ρ_1 and ρ_2 with the support

     $$\{p_{s,t} \mid (s,t) \in \{i_{1,1},\, i_{1,2},\, i_{2,2}\} \times [n] \cup [n+1] \times \{j_1,\, j_2\}\}$$

that substitute zero to all variables with six exceptions: $\rho_1(p_{i_{1,1},j_1}) = 1$, $\rho_1(p_{i_{1,2},j_1}) = 1$, $\rho_1(p_{i_{2,2},j_2}) = 1$, $\rho_2(p_{i_{1,1},j_1}) = 1$, $\rho_2(p_{i_{2,1},j_2}) = 1$, and $\rho_2(p_{i_{2,2},j_2}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-2}_{n-2}$ and thus, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$.

- The third case is when $i_{1,1}$, $i_{1,2} \in \{i_{2,1}, i_{2,2}\}$ and $j_1 \neq j_2$. Without loss of generality $i_{1,1} = i_{2,1}$ and $i_{1,2} = i_{2,2}$. Let us consider substitutions ρ_1 and ρ_2 with the support

$$\{p_{s,t} \mid (s,t) \in \{i_{1,1}, i_{1,2}\} \times [n] \cup [n+1] \times \{j_1, j_2\}\}$$

that substitute zero to all variables with four exceptions: $\rho_1(p_{i_{1,1},j_1}) = 1$, $\rho_1(p_{i_{1,2},j_1}) = 1$, $\rho_2(p_{i_{1,1},j_2}) = 1$, $\rho_2(p_{i_{1,2},j_2}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-1}_{n-1}$ and thus, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$.

- The fourth case is when $i_{1,1} \notin \{i_{2,1}, i_{2,2}\}$, $i_{1,2} \in \{i_{2,1}, i_{2,2}\}$, and $j_1 = j_2$. Without loss of generality $i_{1,2} = i_{2,1}$. Let us fix any $j \in [n] \setminus \{j_1\}$ and consider substitutions ρ_1 and ρ_2 with the support

$$\{p_{s,t} \mid (s,t) \in \{i_{1,1}, i_{1,2}, i_{2,2}\} \times [n] \cup [n+1] \times \{j_1, j\}\}$$

that substitute zero to all variables with six exceptions: $\rho_1(p_{i_{1,1},j_1}) = 1$, $\rho_1(p_{i_{1,2},j_1}) = 1$, $\rho_1(p_{i_{2,2},j}) = 1$, $\rho_2(p_{i_{1,1},j}) = 1$, $\rho_2(p_{i_{1,2},j_1}) = 1$, and $\rho_2(p_{i_{2,2},j_1}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-2}_{n-2}$ and thus, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$.

- The fifth case is when $i_{1,1}, i_{1,2} \notin \{i_{2,1}, i_{2,2}\}$ and $j_1 = j_2$. Let us fix any $j_3, j_4 \in [n] \setminus \{j_1\}$ and consider substitutions ρ_1 and ρ_2 with the support

$$\{p_{s,t} \mid (s,t) \in \{i_{1,1}, i_{1,2}, i_{2,1}, i_{2,2}\} \times [n] \cup [n+1] \times \{j_1, j_3, j_4\}\}$$

that substitute zero to all variables with eight exceptions: $\rho_1(p_{i_{1,1},j_1}) = 1$, $\rho_1(p_{i_{1,2},j_1}) = 1$, $\rho_1(p_{i_{2,1},j_3}) = 1$, $\rho_1(p_{i_{2,2},j_4}) = 1$, $\rho_2(p_{i_{1,1},j_3}) = 1$, $\rho_2(p_{i_{1,2},j_4}) = 1$, $\rho_2(p_{i_{2,1},j_1}) = 1$, and $\rho_2(p_{i_{2,2},j_1}) = 1$. Note that the Boolean function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is equivalent to $\mathtt{PHP}^{n-3}_{n-3}$ and thus, by Corollary 3.17, the π-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ has size $2^{\Omega(n)}$. ⊣

To show that the lower bounds on $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ imply the lower bounds on either F_1 or F_2 we rely on the fact that the orders of the variables are the same in F_1 and F_2. This requirement may look artificial, but it is necessary to use it in order to prove a lower bound for the pigeonhole principle. If it is allowed to use the conjunction rule to OBDDs in different orders, there is the following polynomial size refutation of $\mathtt{PHP}^{n+1}_n$. First, we compute the conjunction of all the long clauses using the pigeon-wise order, i.e., the order $p_{1,1}, p_{1,2}, \dots, p_{1,n}, \dots, p_{n+1,n}$. Afterwards, we compute the conjunction of all the short clauses using the hole-wise order; i.e., the order $p_{1,1}, p_{2,1}, \dots, p_{n+1,1}, \dots, p_{n+1,n}$. Finally, we compute the conjunction of these diagrams and get a contradiction.

§4. OBDD($\wedge$, reordering) is stronger than OBDD($\wedge$)**.** In this section we give an example of a family of unsatisfiable formulas Φ_n that have OBDD($\wedge$, reordering) proofs of polynomial size while all OBDD($\wedge$) proofs of it have size at least $2^{\Omega(n)}$.

Theorem 4.1. *Let $\Psi_n(x_1, x_2, \dots, x_n)$ be a family of unsatisfiable formulas of size poly(n) that satisfies the following conditions:*

- *there exists an order τ such that Ψ_n has a τ-*OBDD($\wedge$) *refutation of size poly(n);*
- *there exists a polynomial $p(n)$, an integer $k \leq \log(p(n))$ and permutations $\sigma_1, \sigma_2, \dots, \sigma_{2^k} \in S_n$ such that for any permutation $\pi \in S_n$ there exists $i \in [2^k]$ such that any $\pi\sigma_i$-*OBDD($\wedge$) *proof of Ψ_n has size at least $2^{\Omega(n)}$.*

Then the formula

$$\Phi_n(w_1, w_2, \dots, w_k, x_1, x_2, \dots, x_n) = \bigwedge_{i=1}^{2^k} \left((w = i-1) \to \Psi_n(x_{\sigma_i(1)}, x_{\sigma_i(2)}, \dots, x_{\sigma_i(n)})\right)$$

has an OBDD($\wedge$, reordering) *proof of size poly(n), but any* OBDD($\wedge$) *proof has size at least $2^{\Omega(n)}$. Here the equality $w = i-1$ means that $w_1 w_2 \dots w_k$ is a binary representation of $i-1$. We assume that Φ_n is written in CNF as follows: we add a clause $\neg(w = i-1)$ to every clause of $\Psi(x_{\sigma_i(1)}, x_{\sigma_i(2)}, \dots, x_{\sigma_i(n)})$.*

Let us consider the following sketch of the proof, before we give a complete proof of this theorem. The lower bound. Consider an OBDD($\wedge$) proof T of the formula Φ_n, let τ be the order of the variables $x_1, x_2, \dots, x_n$ that is induced by the order from T. By the statement of the theorem, there exists $i \in [2^k]$ such that all $(\tau\sigma_i)$-OBDD($\wedge$) proofs of Ψ_n have at least the exponential size. We make a substitution $w = i-1$ to the proof T. This substitution converts the proof of Φ_n into a proof of Ψ_n with the order $\tau\sigma_i$. Hence, T has exponential size.

The upper bound. Since there exists a polynomial sized OBDD($\wedge$) proof of Ψ_n, then for all i there is an order π_i (we may assume after the permutation π_i the variables w get the leftmost positions) such that there is a π_i-OBDD($\wedge$) derivation of the diagram representing $w \neq i-1$ from the ith part of the conjunction. So for each $i \in [2^k]$ we can derive π_i-OBDD representing $w \neq i-1$, and using the reordering rule, we can derive a contradiction.

Proof. The lower bound. Consider an α-OBDD($\wedge$) proof T of the formula Φ_n for some order α. We consider α as a permutation of $[n+k]$. Let π be a permutation of $[n]$ (or variables $x_1, x_2, \dots, x_n$) that is induced by α (formally $\pi(j) = |\{i \in [n] \mid \alpha(i+k) \leq \alpha(j+k)\}|$).

By the statement of the theorem there exists an $i_0 \in [2^k]$ such that any $\pi\sigma_{i_0}$-OBDD($\wedge$) proof of Ψ_n has size at least $2^{\Omega(n)}$. Consider a partial substitution ρ that assign to the variables $w_1, w_2, \dots, w_k$ a binary representation of $i_0 - 1$. Consider the formula $\Phi_n|_\rho$. Clauses that represent $(w = i-1) \to \Psi_n(x_{\sigma_i(1)}, x_{\sigma_i(2)}, \dots, x_{\sigma_i(n)})$ for $i \neq i_0$ are satisfied by ρ. Hence, $\Phi_n|_\rho$ is equal to $\Psi(x_{\sigma_{i_0}(1)}, x_{\sigma_{i_0}(2)}, \dots, x_{\sigma_{i_0}(n)})$. Thus, if we substitute ρ to the proof T we get a π-OBDD($\wedge$) proof $\Psi(x_{\sigma_{i_0}(1)}, x_{\sigma_{i_0}(2)}, \dots, x_{\sigma_{i_0}(n)})$; this proof can be seen as a $\pi\sigma_{i_0}$-OBDD($\wedge$) proof of $\Psi(x_1, x_2, \dots, x_n)$ that has size at least $2^{\Omega(n)}$ by choice of i_0. Thus, T has size at least $2^{\Omega(n)}$.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

The upper bound. Let τ be an order of variables $x_1, x_2, \dots, x_n$ such that $\Psi_n(x_1, x_2, \dots, x_n)$ has a short τ-OBDD($\wedge$) proof.

We describe a short OBDD($\wedge$, reordering) proof of Φ_n. Let μ_i orders the variables w_1, w_2, ..., w_k, x_1, x_2, ..., x_n such that x_1, x_2, ..., x_n are ordered by $\tau\sigma_i^{-1}$ and the variables $w_1, w_2, \dots, w_k$ appear before the variables x_1, x_2, ..., x_n. In other words, μ_i orders variables as follows: $w_1, w_2, \dots, w_k, x_{\tau\sigma_i^{-1}(1)}, x_{\tau\sigma_i^{-1}(2)}, \dots, x_{\tau\sigma_i^{-1}(n)}$.

Consider a polynomial sized τ-OBDD($\wedge$) proof of Ψ_n. It is easy to see that this refutation may be transformed into a μ_i-OBDD($\wedge$) derivation of a diagram that represents $w \neq i$ from the formula $(w = i - 1) \to \Psi(x_{\sigma_i(1)}, x_{\sigma_i(2)}, \dots, x_{\sigma_i(n)})$. Indeed, the variables $w_1, w_2, \dots, w_k$ in the order μ_i appear in the beginning, hence, every diagram D from the original proof will be transformed to a diagram that represents $D \vee \neg(w = i)$ and the latter diagram has size at most $|D| + O(k)$, where $|D|$ is size of the diagram D.

So we have all diagrams that represent $w \neq i - 1$ for all $i \in [2^k]$. Formally these diagrams use different orders μ_i but in fact the diagrams depend essentially only on the variables $w_1, w_2, \dots, w_k$, and all μ_i order them in the same way. Thus, if we change orders in all of these diagrams to some "standard" one, the diagrams will be not changed. Then we apply the conjunction rule to these diagrams and get a constant false diagram since $w_1 w_2 \dots w_k$ is a binary representation of $i - 1$ for some $i \in [2^k]$. All intermediate diagrams have size at most $2^k poly(n)$. Hence, we get a proof of size poly(n). ⊣

Now we construct the family of unsatisfiable formulas Ψ_n that satisfies the conditions of Theorem 4.1. We use an argument similar to the proofs of the lower bounds for OBDD($\wedge$, reordering) proofs. At first, we construct a function that has sizes of OBDD representations in different orders close to the required sizes of proofs for Ψ_n.

Proposition 4.2. *In the order $x_1, x_{n+1}, x_2, x_{n+2}, \dots, x_n, x_{2n}$ the function $\mathtt{EQ}_n$ has an* OBDD *representation of size $3n + 2$.*

Proof. The proof can be easily done by induction, using the following equation: $\mathtt{EQ}_n(x_1, x_2, \dots, x_n, x_{n+1}, \dots, x_{2n}) = (x_1 = x_{n+1}) \wedge \mathtt{EQ}_{n-1}(x_2, \dots, x_n, x_{n+2}, \dots, x_{2n})$. ⊣

The proof of the following lemma is similar to the proof of the $\Omega(n)$ lower bound on the best communication complexity of the shifted equality function [20, Example 7.9].

Lemma 4.3. *Let σ_i for $i \in [n]$ be a permutation such that $\sigma_i(j) = j$ for $j \in [n]$ and $\sigma_i(n+j) = n + (i + j - 1 \bmod n) + 1$ for $j \in [n]$. Let Π be a balanced partition of $[2n]$, i.e., $\Pi = (\Pi_0, \Pi_1)$ and $|\Pi_0| = |\Pi_1| = n$. Then there exists $i \in [n]$ such that communication complexity of $\mathtt{EQ}_n$ with respect to $\sigma_i\Pi$ is at least $\Omega(n)$, where*

$$\Pi\sigma_i = (\{j \mid \sigma_i(j) \in \Pi_0\}, \{j \mid \sigma_i(j) \in \Pi_1\}).$$

Proof. Let $\Gamma = (\Gamma_0, \Gamma_1)$ be a balanced partition of $[2n]$, $V_x^\Gamma = \{i \in [n] \mid i \in \Gamma_0\}$, and $V_y^\Gamma = \{i \in [n] \mid (i + n) \in \Gamma_0\}$.

We denote by $B^\Gamma = V_x^\Gamma \oplus V_y^\Gamma$ the symmetric difference of V_x^Γ and V_y^Γ.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Claim 4.3.1 *The communication complexity of* $\mathtt{EQ}_n$ *with respect to a partition* Γ *is at least* $|B^\Gamma|$. ⊣

Proof. Let $V_x^\Gamma \setminus V_y^\Gamma = \{i_1,\dots,i_{\ell_1}\}$ and $V_y^\Gamma \setminus V_x^\Gamma = \{j_1,\dots,j_{\ell_2}\}$. Note that the communication complexity of $\mathtt{EQ}_{\ell_1+\ell_2}$ with respect to a partition $([1,\dots,\ell_1+\ell_2],[\ell_1+\ell_2+1,2(\ell_1+\ell_2)])$ is at least $\ell_1+\ell_2=|B^\Gamma|$ by Lemma 2.4. Additionally, if we define

$$v_k=\begin{cases} x_{k'} & \text{if } k=i_{k'},\\ x_{k'} & \text{if } k=j_{k'}+n,\\ x_{k'+\ell_1+\ell_2} & \text{if } k=i_{k'}+n,\\ x_{k'+\ell_1+\ell_2} & \text{if } k=j_{k'},\\ 0 & \text{otherwise}\end{cases}$$

for $k\in[2n]$, then $\mathtt{EQ}_n(v_1,\dots,v_{2n})=\mathtt{EQ}_{\ell_1+\ell_2}(x_1,\dots,x_{\ell_1+\ell_2},\dots,x_{2(\ell_1+\ell_2)})$. Since Alice can compute v_i for $i\in\Gamma_0$ and Bob can compute v_j for $j\in\Gamma_1$, we prove that communication complexity of $\mathtt{EQ}_n$ with respect to a partition Γ is at least $\ell_1+\ell_2=|B^\Gamma|$. ⊣

If for some $i\in[n]$ size $|B^{\Pi\sigma_i}|\geq\frac{n}{32}$, then by Claim 4.3.1, the communication complexity of $\mathtt{EQ}_n$ with respect to $\Pi\sigma_i$ is at least $n/32$. Assume that for all i the following holds: $|B^{\Pi\sigma_i}|<\frac{n}{32}$.

Let us show that $|V_y^\Pi|\geq\frac{15n}{32}$. Indeed, we know that $|V_x^\Pi|+|V_y^\Pi|=n$; hence, $|V_x^\Pi\cup V_x^\Pi|\geq\frac{n}{2}$. Since σ_n is the identical permutation, we get $n/32>|V_x^\Pi\oplus V_y^\Pi|=|V_x^\Pi\cup V_y^\Pi|-|V_x^\Pi\cap V_y^\Pi|\geq\frac{n}{2}-|V_y^\Pi|$, and therefore $|V_y^\Pi|\geq\frac{15n}{32}$.

By definition, $B^{\Pi\sigma_i}=V_x^{\Pi\sigma_i}\oplus V_y^{\Pi\sigma_i}$. Since the permutation σ_i does not move variables x_j for $j\in[n]$, we get $V_x^{\Pi\sigma_i}=V_x^\Pi$ for all $i\in[n]$. Note that if $|V_x^\Pi\oplus V_y^{\Pi\sigma_i}|<n/32$ and $|V_x^\Pi\oplus V_y^{\Pi\sigma_j}|<n/32$, then $|V_y^{\Pi\sigma_i}\oplus V_y^{\Pi\sigma_j}|<n/16$ for all $i,j\in[n]$.

We show that there exists a $j\in[n]$ such that $|V_y^\Pi\oplus V_y^{\Pi\sigma_j}|\geq n/16$. The latter would be a contradiction with our assumption since σ_n is an identical substitution. Consider the sum $\sum_{k=1}^{n}|V_y^\Pi\cap V_y^{\Pi\sigma_k}|$. On the one hand this sum equals $|V_y^\Pi|^2$. Indeed,

$$\begin{aligned}\sum_{k=1}^{n}|V_y^\Pi\cap V_y^{\Pi\sigma_k}| &= \sum_{k=1}^{n}\sum_{i\in V_y^\Pi}\mathbf{1}_{i\in V_y^{\Pi\sigma_k}}\\ &=\sum_{k=1}^{n}\sum_{i\in V_y^\Pi}\mathbf{1}_{\sigma_k(i+n)\in\Pi_0}=\sum_{k=1}^{n}\sum_{i=V_y^\Pi}\mathbf{1}_{\sigma_k(i+n)-n\in V_y^\Pi}\\ &=\sum_{i\in V_y^\Pi}\sum_{j\in V_y^\Pi}|\{k\in[n]\mid\sigma_k(i+n)-n=j\}|=\sum_{i\in V_y^\Pi}\sum_{j\in V_y^\Pi}1=|V_y^\Pi|^2.\end{aligned}$$

On the other hand

$$\sum_{k=1}^{n}|V_y^\Pi\cap V_y^{\Pi\sigma_k}|\geq\sum_{k=1}^{n}\Big(|V_y^\Pi\cup V_y^{\Pi\sigma_k}|-|V_y^\Pi\oplus V_y^{\Pi\sigma_k}|\Big)\geq$$

$$\sum_{k=1}^{n}\Big(|V_y^\Pi|-|V_y^\Pi\oplus V_y^{\Pi\sigma_k}|\Big)>n|V_y^\Pi|-\frac{n^2}{16}.$$

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Note that if $|V_y^{\Pi}| \geq \frac{n}{2}+\frac{n}{64}$, then $|V_x^{\Pi}| < \frac{n}{2}-\frac{n}{64}$ and $|V_x^{\Pi} \oplus V_y^{\Pi\sigma_i}| \geq \frac{n}{32}$ which contradicts the assumption. If $|V_y^{\Pi}| < \frac{n}{2}+\frac{n}{64}$, then $|V_y^{\Pi}|^2 \leq \frac{n^2}{4}+\frac{n^2}{32}$ but $n|V_y^{\Pi}| - \frac{n^2}{16} \geq \frac{15n^2}{32} - \frac{n^2}{16}$ and this is a contradiction.

Corollary 4.4. *Let σ_i for $i \in [n]$ be a cyclic permutation of variables y that maps y_j to $y_{i+j \bmod n+1}$ for any $j \in [n]$. Formally $\sigma_i(j) = j$ for $j \in [n]$ and $\sigma_i(n+j) = n+(i+j-1 \bmod n)+1$ for $j \in [n]$. Then for any order π on $2n$ variables there exists $i \in [n]$ such that every $\pi\sigma_i$-OBDD representation of $\mathtt{EQ}_n$ has size at least $2^{\Omega(n)}$.*

Now we are ready to construct a formula that may be used in Theorem 4.1. Consider a formula $\Psi_n(x,y,z)$ from $3n+1$ variables (here x,y are vectors of n variables and z is a vector of $n+1$ variables) that is the conjunction of CNF representations of the following conditions:

- $x_i = y_i$ for all $i \in [n]$;
- z_0;
- $(x_i = y_i) \to (z_{i-1} \to z_i)$ for all $i \in [n]$;
- $\neg z_n$.

Note that $\Psi_n(x,y,z)$ is unsatisfiable since we have that $x_i = y_i$ for all i; it implies that $z_i = 1$ for all i, but z_n should be zero. The following statement is straightforward.

Proposition 4.5. *$\Psi_n(x,y,z)$ has an OBDD$(\wedge)$ refutation of polynomial size using the following order over the variables of $\Psi_n(x,y,z)$: $z_0, x_1, y_1, z_1, x_2, y_2, z_2, \dots, x_n, y_n, z_n$.*

Proposition 4.6. *$\Psi_n(x,y,z)$ is minimally unsatisfiable, i.e., it becomes satisfiable after removing any of its clauses.*

Proof. We consider 3 cases.

1. If we remove the clause $\neg z_n$ (or z_0), then identically 1 (or identically 0) assignment satisfies all other clauses.
2. If we remove a clause that represents $x_i = y_i$ (w.l.o.g. assume that this is the clause $\neg x_i \vee y_i$), then we assign $x_j = 1$ for all j, $y_j = 1$ for all $j \neq i$, $y_i = 0$ and $z_j = 1$ for $j < i$ and $z_j = 0$ for $j \geq i$.
3. If we remove a clause that represents $(x_i = y_i) \to (z_{i-1} \to z_i)$ (w.l.o.g. assume that this is the clause $x_i \vee y_i \vee \neg z_{i-1} \vee z_i$), then we assign $x_j = y_j = 0$ for all $j \in [n]$ and $z_j = 1$ for $j < i$ and $z_j = 0$ for $j \geq i$. ⊣

Lemma 4.7. *For any order π on the variables x, y, and z the size of a π-OBDD$(\wedge)$ proof of Ψ_n is at least $\frac{1}{10}\sqrt{S}$, where S is the size of the shortest π'-OBDD representation of $\mathtt{EQ}_n(x,y)$, where π' is the order induced by π on the variables x and y.*

Proof. Assume that in the last step of the proof we apply the conjunction operation to F_1 and F_2 and get a contradiction. By Proposition 4.6, Ψ_n is minimal unsatisfiable; hence, every clause of Ψ_n was joined either to F_1 or to F_2. F_1 and F_2 are not identically false functions. By the argument similar to the proof of Proposition 4.6 there exists a satisfying assignment τ_1 of F_1 and there exists a set $I_1 \subseteq [n]$ of size at least $(n-1)$ such that for all $i \in I_1$ the values of variables $\tau_1(x_i) = \tau_1(y_i)$. Moreover, if for arbitrary set $J \subseteq I_1$ we simultaneously change τ_1-values of variables x_j and y_j for $j \in J$, then τ_1 remains a satisfying assignment of F_1. Let ρ_1 be a partial substitution that does not substitute values for x_i and y_i for $i \in I$ and for all other variables it is consistent with τ_1. We define I_2 and ρ_2 analogously. Let us consider the function $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$; it contains all clauses of $\mathtt{EQ}_n(x, y)$ that have occurrences of x_i and y_i for $i \in I_1 \cap I_2$ and possibly several remaining clauses of $\mathtt{EQ}_n(x, y)$. Hence, $\mathtt{EQ}_n(x, y)$ can be obtained from $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ by addition of at most four clauses, and each of them depends on two variables. Hence, S (recall that S is the shortest π'-OBDD representation of $\mathtt{EQ}_n(x, y)$) is at most 100 times the size of any π'-OBDD representation of $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$. Therefore, an π-OBDD-representation of either F_1 or F_2 has size at least $\frac{1}{10}\sqrt{S}$. ⊣

Theorem 4.8. *There exists an unsatisfiable CNF formula Φ_n of size poly(n) such that there exists a polynomial size* OBDD$(\wedge,$ reordering$)$ *proof of Φ_n but every* OBDD$(\wedge)$ *proof of Φ_n has size $2^{\Omega(n)}$.*

Proof. By Lemma 4.7 for every order π the size of π-OBDD$(\wedge)$ proof of Ψ_n is at least $\frac{1}{10}\sqrt{S}$, where S is the size of the shortest π'-OBDD representation of $\mathtt{EQ}_n$. By Corollary 4.4 there exists a family of permutations σ_i of $[2n]$ for $i \in [n]$ such that for every order τ there exists an $i \in [n]$ such that size of any $\tau\sigma_i$-OBDD$(\wedge)$ proof of Ψ_n is at least $2^{\Omega(n)}$. By Proposition 4.5 there exists an order τ and a τ-OBDD$(\wedge)$ proof of Ψ_n of size poly(n). Theorem 4.1 gives a construction of the desired formula Φ_n. ⊣

§5. OBDD$(\wedge, \exists,$ reordering$)$ algorithms.

It is known that OBDD$(\wedge, \exists)$ algorithms can solve $\mathtt{PHP}_n^{n+1}$ in polynomial time [7]. Now we show that the Tseitin formulas are also easy for OBDD$(\wedge, \exists)$ algorithms.

Proposition 5.1. *There exists an* OBDD$(\wedge, \exists)$ *algorithm that solves Tseitin formulas in polynomial time.*

Informally the proof of this theorem is as follows. Notice that the projection of two linear equations over the common variable is just the sum of these equations. Since each variable of a Tseitin formula occurs exactly twice, we can sum up all equations in every connected component.

Proof. Consider the following OBDD$(\wedge, \exists)$ algorithm.

1. If $S \neq \emptyset$, choose $C \in S$ and apply the join operation; otherwise goto Step 5.
2. If D depends on some variable x, apply the join operation to all clauses in S that depend on x.
3. Apply the projection operation over the variable x.
4. If D represents a constant then go to Step 1, otherwise goto Step 2.
5. If D is satisfiable then return 1, otherwise return 0.

We prove that for arbitrary order on variables this algorithm solves Tseitin formulas in polynomial-time. We note that after Step 2 the current diagram D represents the conjunction of two linear equations and after Step 3 the current diagram D represents one linear equation. Indeed, every variable of a Tseitin formula has occurrences in exactly two equations, and the projection of two linear equations over the common variable is just the sum of this equations.

It is easy to see that the algorithm chooses a connected component of the graph and sums up all equations from this component. If the component is unsatisfiable, the algorithm stops and return 0; otherwise it goes to the next component. Finally,

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

if all components are satisfiable, it returns 1. We have to show that D always has polynomial size.

Any OBDD that represents one linear equation have linear size. Consider some equation from the input Tseitin formula that depends on d variables, the maximum size of OBDD for the subset of clauses has size at most $2^{d+1}-1$ (the size of a decision tree). The size of any CNF representation of a linear equation is at least 2^{d-1}. Hence, the OBDD size of conjunction of the subset of clauses is less than the size of the input formula multiplied by 4. We already know that D represents either a linear equation, or a conjunction of several clauses representing a parity condition of a vertex, or the conjunction of previous two. In all of this cases size of D is bounded by a polynomial in the size of the input formula. Since all operations with OBDD can be performed in polynomial time, the running time of the algorithm is polynomial in the size of the input formula. ⊣

The following theorem implies that every OBDD for a characteristic function of a good enough code has at least exponential size.

Theorem 5.2. *Let $C \subseteq \{0,1\}^n$ be a $(\frac{1}{2}+\varepsilon, L)$-erasure list-decodable code. Then for every partition Π such that $|\Pi_0| = \frac{n}{2}$, the communication complexity of the characteristic function of C (i.e., function $\chi_C \colon \{0,1\}^n \to \{0,1\} : \forall c \in \{0,1\}^n\ \chi_C(x) = 1 \Leftrightarrow c \in C$) with respect to Π is at least $\log \frac{|C|}{L^2}$.*

Moreover, for every tuple of k different indices $i_1, \dots, i_k \in [n]$ $(0 \le k \le 2\varepsilon n)$ and every partition Π such that $|\Pi_0| = \frac{n-k}{2}$, the communication complexity of the Boolean function $\exists x_{i_1} \dots \exists x_{i_k}\ \chi_C(x_1, \dots, x_n)$ with respect to Π is at least $\log \frac{|C|}{L^2}$.

Proof. It is enough to prove the "moreover" part of the statement since the first part is its special case (with $k=0$). Without loss of generality we may assume that $\Pi = (\{1, \dots, \frac{n-k}{2}\}, \{\frac{n-k}{2}+1, \dots, n-k\})$ and $i_j = n-k+j$ for all $j \in [k]$.

We prove that any 1-monochromatic rectangle for $\chi(x_1, \dots, x_{n-k}) = \exists y_1, \dots, y_k\ \chi_C(x_1, \dots, x_{n-k}, y_1, \dots, y_k)$ contains at most L^2 elements. Let $R = A \times B$ be some 1-monochromatic rectangle and $a \in A$. Consider the string $c = a?^{n-|a|}$: it has $n - \frac{n-k}{2} = \frac{n+k}{2} \le (\frac{1}{2}+\varepsilon)n$ symbols ?, hence, there are at most L codewords that are consistent with c. As a result, $|B| \le L$. It is clear that the same argument implies that $|A| \le L$.

Hence, there are at least $\frac{|C|}{L^2}$ 1-monochromatic rectangles and, thus, by Lemma 2.3, the communication complexity of χ with respect to Π is at least $\log \frac{|C|}{L^2}$. ⊣

Theorem 5.3. *Let $C \subseteq \{0,1\}^n$ be a linear code with the relative distance $\frac{1}{3}$ such that the checksum matrix H of the code C has the following properties:*

- *H is a binary matrix of size $\alpha n \times n$, where $\alpha \in (0,1)$ is a constant;*
- *every row of H contains at most $t(n)$ ones, where t is some function;*
- *in every $\frac{1}{6}n$ columns of H we can find ones in at least $(\alpha - \delta)n$ different rows of the matrix, where $\delta \in \left(0, \min\{\alpha, \frac{1-\alpha}{2}\}\right)$ is a constant.*

Denote by F_n the canonical CNF representation of the system of linear equations $H(x) = 0$; F_n is in $t(n)$-CNF, hence the size of F_n is at most $\alpha n 2^{t(n)-1} t(n)$. Then every OBDD($\wedge, \exists$, reordering) *algorithm runs on the formula F_n for at least $2^{\Omega(n)}$ steps.*

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

Proof. Since the code C has relative distance $\frac{1}{3}$, By Theorem 2.6 applied to $\varepsilon = 1/4$, C is a $(\frac{7}{12}, 8)$-erasure list-decodable code. We apply an OBDD$(\wedge, \exists)$ algorithm on the formula F_n. We argue that during the execution of the algorithm, the size of the diagram D reaches at least $2^{\Omega(n)}$.

Assume that during the execution of the algorithm we apply the projection operation at least $k = \frac{1}{6}n$ times. Let us consider the diagram D after the moment when the algorithm applies the projection operation exactly k times. Let D represent at this moment a function of type $\exists y_1, \dots, y_k\ \phi(x_1, \dots, x_{n-k}, y_1, \dots, y_k)$, where ϕ is the conjunction of several clauses from F_n. The projection operation on a variable x can be applied only if all clauses from S do not depend on x. Therefore, all clauses corresponding to linear equations with the variable x must be among clauses of ϕ. By the assumption of the theorem, any k columns of H have ones in at least $(\alpha - \delta)n$ rows. Thus, ϕ contains all clauses from the representation of $(\alpha - \delta)n$ equations and possibly several other clauses from other equations.

Lemma 2.7 implies that ϕ is a characteristic function of a $(\frac{7}{12}, 8 \cdot 2^{\delta n})$-erasure list-decodable code. The number of satisfying assignments of ϕ is at least the number of solutions of the system $Hx = 0$. Hence, the size of the code defined by ϕ is at least $2^{(1-\alpha)n}$. Since $k \le \frac{1}{6}n$, by Theorem 5.2 and Lemma 2.1, the size of every OBDD for the formula $\exists y_1, \dots, y_k\ \phi(x_1, \dots, x_{n-k}, y_1, \dots, y_k)$ is at least $2^{(1-\alpha)n} 2^{-2\delta n} \frac{1}{16} = 2^{(1-\alpha-2\delta)n-4}$, whatever is the order of variables.

The case where the algorithm applies the projection operator less than $n/6$ times is similar: we should consider the last diagram D in the execution of the algorithm. ⊣

Theorem 5.3 claims that OBDD$(\wedge, \exists, \text{reordering})$ algorithms run in exponential time on formulas F_n representing error correcting codes with some specific properties. At this point, we have not yet proven the existence of codes matching the requirements of the theorem. In what follows we show that such linear codes do exist. In Section 6.1 we prove the existence of suitable codes by a probabilistic argument; in Section 6.2 we prove a stronger statement and show that the required codes can be constructed explicitly. These constructions of codes together with Theorem 5.3 imply the following result:

Corollary 5.4. *For all large enough n there exists a CNF formula with n Boolean variables, of size $O(n)$ such that every* OBDD$(\wedge, \exists, \text{reordering})$ *algorithm runs on this formula at least $2^{\Omega(n)}$ steps. Moreover, for a given n such a formula can be constructed by a deterministic algorithm in time poly(n).*

§6. Constructions of suitable linear codes.

6.1. A randomized construction of suitable linear codes. We are going to combine Theorem 5.3 with some specific constructions of codes. In this section we describe a construction of low-density parity codes (LDPC) that can be combined with Theorem 5.3. The idea of low-density parity codes goes back to Gallager [12]. The LDPC are linear codes whose checksum matrix are “sparse”, i.e., contains few ones in each row and each column. We employ the standard properties of the LDPC—a randomly chosen matrix (with suitable parameters) with high probability defines a linear code with large enough distance. We also employ a somewhat less conventional

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

property of LDPC: most of these codes share some property of “uniformity”; that is, even a rather small set of variables must be involved in almost all checksums of the code.

First of all we recall the classic construction of random LDPC from [12]. Let us fix some integer parameters t, r, and n (assuming that t divides n). Define the “basic” matrix A of size $(n/t)\times n$ as a concatenation of t copies of the identity matrix $(n/t)\times(n/t)$:

$$A = \left(\underbrace{ \underbrace{\begin{matrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{matrix}}_{(n/t)\times(n/t)\text{–identity matrix}} \begin{matrix} \\ \\ \\ \dots & \dots & \dots \\ \\ \end{matrix} \underbrace{\begin{matrix} 1 & 0 & 0 & \dots & 0 \\ 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \dots & 1 \end{matrix}}_{(n/t)\times(n/t)\text{–identity matrix}} }_{t\text{ copies}} \right)$$

Notice that each column of A contains one nonzero element; in each row of A there are exactly t nonzero elements. Further, we consider the family of all matrices of size $(rn/t)\times n$ that consist of r horizontal “blocks” of size $(n/t)\times n$, where each block is obtained as a permutation of columns of A,

$$\begin{pmatrix} [\text{ 1st permutation of columns of } A] \\ [\text{2nd permutation of columns of } A] \\ \vdots \\ [r\text{th permutation of columns of } A] \end{pmatrix}$$

It is easy to see that in each column of this matrix there are r ones, and in each row there are exactly t ones. We introduce the uniform distribution on all matrices of this type. We can interpret these matrices as checksums matrices of some linear codes. Gallager proved that most of these codes have rather large minimal distance.

Proposition 6.1 see [12]. *Most (say, at least 90%) of matrices in Gallager's family define a linear code with parameters approaching Gilbert–Varshamov bound. More precisely, for every $\delta \in (0,\frac{1}{2})$ and for every t there exists $r = r(t)$ such that for large enough n most matrices from the defined family have minimal distance $\geq \delta n$. Moreover, the ratio $r(t)/t$ approaches $h(2\delta)$ as t goes to infinity, where $h(x) = -x\log x - (1-x)\log(1-x)$ (the binary entropy). This means that the rate of the code can be made arbitrarily close to $1-h(2\delta)$ (i.e., to the Gilbert–Varshamov bound).*

A family of linear codes defined above can be specified by parameters r,t,n. However, it is more convenient to specify these codes by another triple of numbers—by (δ,t,n) (assuming that the value $r = r(\delta,t,n)$ is defined implicitly as the minimal integer such that most codes of the family have minimal distance greater than δn). Now we can state the main technical lemma of this section.

Lemma 6.2. *For all $\beta \in (0,1)$, $\gamma < 1$, and $\delta \in (0, \frac{1}{2})$, for all large enough t most (say, at least 90%) of linear codes from Gallager's family with parameters (δ, t, n) satisfy the following property: every βn columns in the checksum matrix of the code intersect at least a fraction γ of all rows of the matrix.*

Proof. By construction, every checksum matrix from Gallager's codes family consists of r blocks of size $(n/t) \times n$ (where each if these blocks is a permutation of columns of the basic matrix A defined above). Let us fix some real $\rho > 0$. We say that some tuple of βn columns is *poor* for some block of the matrix, if these columns intersect with at most $\rho \cdot (n/t)$ rows of the block. We will proceed as follows.

Step 1. We estimate the probability of the event that one fixed tuple of βn columns is poor for a randomly chosen block (which is obtained as a random permutation of columns in A).

Step 2. We estimate the probability that one fixed tuple of βn columns is poor for more than ρr randomly chosen blocks.

Step 3. At last, we estimate the probability that *at least one* tuple of βn columns is poor for more than ρr (independently) randomly chosen blocks.

We are going to show that the probability estimated on Step 3 becomes very small for a suitable t. It follows that for most Gallager's codes, *every* tuple of βn columns intersects all but $2\rho rn/t$ rows (one fraction of ρ rows comes from all blocks where these columns are poor, and another fraction of ρ rows comes from all other blocks). We can choose ρ so that $1 - 2\rho > \gamma$, and the Lemma will follow. Thus, it remains to bound the probabilities in Steps 1–3.

Step 1: We fix some tuple of βn indices of columns, then take a random permutation of columns in a matrix A and count the total number of intersected rows. These random procedure can be equivalently explained in other words: we fix the original matrix A and choose at random a tuple of βn columns. By the construction of the matrix A, every column from the chosen tuple intersects with only one row. We need to estimate the number of rows that intersect with at least one of these βn columns. We have to show that for $\beta n \gg n/t$ a "typical" tuple of columns $\langle i_1, \dots, i_{\beta n}\rangle$ intersects with almost all rows. Let us estimate the probability of the "bad" event, when the randomly chosen columns intersect with only $s \leq (1-\rho)(n/t)$ rows. This bad event means that we can select s positions in the sequence of indices $1, \dots, \beta n$ so that

(i) the s columns with "selected" indices $i_{l_1}, \dots, i_{l_s}$ intersect pairwise different rows of the matrix A, and

(ii) each of the other $(\beta n - s)$ columns intersects once again the same row as one of the s "selected" columns.

Probability of this event is not greater than

$$\sum_{s=1}^{(1-\rho)n/t} \binom{\beta n}{s} \Pr[\text{columns } i_{l_1}, \dots, i_{l_s} \text{ intersect pairwise different rows}](1-\rho)^{\beta n - s} \leq$$

$$\sum_{s=1}^{(1-\rho)n/t} \binom{\beta n}{s} (1-\rho)^{\beta n - s}. \qquad (1)$$

Indeed, for each s we have $\binom{\beta n}{s}$ ways to choose s indices in the list $1,\dots,\beta n$. When the positions of the s "selected" columns are fixed, every next randomly chosen column intersects with one of the previously counted rows with a probability less than

$$\frac{[\text{columns intersecting with given } s \text{ rows}]}{[\text{total number of columns}]} = \frac{s\cdot t}{n} \leq 1-\rho.$$

For a fixed ρ and $(n/t) \ll \beta n$, the sum (1) is less than $2^{-c\beta n}$, where $c > 0$ depends on ρ (and not on n).

Step 2: Let us fix some tuple of βn columns. At Step 1 we proved that the probability to get one poor block is $p_n < 2^{-c\beta n}$. Since all r blocks in the checksum matrix are chosen independently, the probability that at least $k = (1-\rho)r$ of them are poor for the fixed columns is less than $\binom{r}{k}\cdot(p_n)^k \leq 2^r \cdot 2^{-c\beta k n}$. As we increase t, the corresponding values of $r = r(t)$ and $k(t) = (1-\rho)r(t)$ grow linearly. It follows that $2^r\cdot 2^{-c\beta kn}$ decreases as $2^{-\Theta(\beta tn)+O(t)}$ (the multiplicative constants hidden in the terms $\Theta(\beta tn)$ and $O(t)$ depend on ρ but not on t).

Step 3: It remains to multiply the probability from Step 2 by the number of all tuples of βn columns. We get the bound $\binom{n}{\beta n}\cdot 2^{-\Theta(\beta tn)+O(t)} = 2^{h(\beta)n+o(n)}\cdot 2^{-\Theta(\beta tn)+O(t)}$ for the probability that at least one tuple of βn columns intersects too few rows. It remains to choose a suitable $t = t(\beta)$ and make this probability less than 0.1. ⊣

Corollary 6.3. *For the distribution defined above, the system of linear equations* $Hx = 0$ *with probability close to* 1 *can be represented as a CNF of size* $O(n)$.

Thus, we obtain a weak version of Corollary 5.4 (without the *moreover* part) for CNF of size $O(n)$ with n Boolean variables. In other words, for every N there exists a CNF formula of size N such that every OBDD($\wedge,\exists$) algorithm runs on this formula at least $2^{\Omega(N)}$ steps. In the next section we prove a similar result for an explicitly constructed code.

6.2. An explicit construction of suitable linear codes. In this section we present an explicit construction of a linear code with the property of list decoding and a check-sum matrix that is sparse and strongly mixing. More technically, we construct explicitly a code that enjoys the properties from Proposition 6.1 and Lemma 6.2 (in the previous section we proved that this property holds with high probability for *randomly chosen* code from Gallager's family of linear codes).

6.2.1. The large-scale scheme of the construction. Our aim is to construct an explicit family of linear codes with the properties discussed in the previous section. To be more specific, we fix a triple of real parameters (α,β,γ) and define four properties of a linear code (more precisely, we define properties of a *checksum matrix* of a linear code).

(G) The number of codewords matching this matrix is equal to $2^{\Omega(n)}$ (the code is *asymptotically good*).

(S) *Sparseness*: every row contains only $O(1)$ ones (the other matrix elements are zeros).

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

(L) *List decoding*: for every set of αn positions in a codeword

$$1 \le i_1 < \cdots < i_{\alpha n} \le n,$$

and for every assignment of bits b_j to these positions, there exist at most $O(1)$ codewords $\bar{x} = (x_1, \dots, x_n)$ corresponding to this checksum matrix such that $x_{i_j} = b_j$ for $j = 1, \dots, \alpha n$.

(M) *Mixing*: For every set of βn columns of the matrix there exist $\ge \gamma n$ rows such that each of these rows meets at least one of the given βn columns at a matrix cell containing 1.

Now we can formulate the main result of this section.

THEOREM 6.4. *For any values of parameters $\alpha > 0$, $\beta > 0$, and $\gamma < 1$ there exists a polynomial-time computable algorithm that finds for inputs n from a dense enough sequence of integers n (technically, for an arithmetic series of n) a (binary) checksum matrix M_n of size $k(n) \times n$ (with n columns and $k(n)$ linearly independent rows) that satisfies the conditions* (G), (S), (L), *and* (M).

Together with Theorem 5.3 this statement implies Corollary 5.4. Indeed, to prove Corollary 5.4 it is enough to take the natural representation of the linear system constructed in Theorem 6.4.

In the proof of Theorem 6.4 we prefer to use instead of (L) another property that can be defined in terms of the code distance:

(L′) *Code distance*: the distance between every two codewords in the code is $\ge (\frac{1}{2} - \delta)n$.

Notice that (L′) implies (L) due to Theorem 2.6. Thus, to prove the theorem it is enough to provide an explicit construction of codes that satisfy (G), (S), (L′), and (M). Also for some technical reason we will need in the proof a stronger version of (M) with some suitable parameter $q > 0$:

(M′) *Strong mixing*: for every set of βn columns there exist $\ge \gamma n$ rows of the matrix such that each row meets at least q of these columns at a matrix cell containing 1.

The rest of this section is organized as follows. In Section 6.2.2 we explain an explicit construction of a code that satisfies (L′). In Section 6.2.3 we explain an explicit construction of a code that satisfies (M′). In Section 6.2.4 we merge these two constructions. At last, in Section 6.2.5 we briefly discuss the explicit construction of algebraic expanders involved in the previous sections.

6.2.2. A code with a very high distance. Our construction is iterative: we start with some asymptotically good explicitly constructed linear code and then amplify its parameters by iterating $O(1)$ times some transformation of this code, as explained below.

The starting point of our construction is an asymptotically good linear code with the code length N and code distance ρN for some constant $\rho > 0$. For instance, we can take the explicit construction of expander codes suggested by Zémor, [30]. So far we only assume that $\rho > 0$ (so the code distance can be much less than $N/2$).

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

In what follows we apply ($O(1)$ times) to this code some amplification procedure. Each iteration of this procedure

- extends the length of codewords by some constant factor,
- does not change the number of codewords,
- makes the code distance closer to $1/2$ of the code length.

The amplification procedure decreases the rate of the code by some constant factor. However, the code remains asymptotically good. After $O(1)$ iteration of this procedure we will obtain a code that satisfies properties (G), (S), and (L$'$). At this stage we cannot guarantee the property (M) (to achieve (M) we will need some more work, see the next section). To "amplify" a code with the length N and the distance ρN we will use an (explicitly constructed) expander graph $G = (V,E)$ on N vertices, with the following property of *strong ε-expansion*: *For every set of vertices $S \subseteq V$ of size $\leq (\frac{1}{2} - \varepsilon)n$ the number of neighbors of this set (i.e., the number of vertices $w \in V$ that are connected by an edge with at least one vertex $v \in S$) is not less than* $(2+\varepsilon)|S|$.

Remark: A graph with the property of strong ε-expansion can be constructed from any algebraic expander as follows. We begin with an algebraic expander with parameters $(N,d,1/2)$, where N is the number of vertices, d is the degree of each vertex, and the parameter $1/2$ means that the second largest eigenvalue of the adjacency matrix is not greater than $d/2$ (see [15] for a survey of known explicit constructions of algebraic expanders). Then we modify this graph by adding to each vertex a loop (an edge that goes from a vertex to itself). At last, we take a suitable degree r of the modified graph. In other words, the edges of the new graph are paths of length $\leq r$ in the initial algebraic expander. The less is the value of ε, the bigger length of the path $r = r(\varepsilon)$ we need to take, and the bigger is the degree of the resulting graph. However, for every constant ε, this construction gives a graph with the property of strong ε-expansion and some degree $D = D(\varepsilon)$, which does not depend on N. We may assume that ε is less than the value of δ from (L$'$).

In what follows we describe the *amplification* procedure. We assume that we are given a linear code $\mathcal{C}$ with the code length N and the code distance ρN. We fix some one-to-one correspondence between the bits of the codewords of $\mathcal{C}$ $(i = 1,\dots,N)$ and the vertices of a graph G (with N vertices, of some degree D) with the property of strong ε-expansion defined above. In what follows we transform $\mathcal{C}$ into a code $\mathcal{C}'$, and then into another code $\mathcal{C}''$. The last one will be the required "amplified" code.

The first stage of the amplification: from $\mathcal{C}$ to $\mathcal{C}'$. We construct a new code $\mathcal{C}'$ where each codeword will consist of $N' := DN$ bits, with the distance $\rho N'$ (i.e., the relative code distance in the new code $\mathcal{C}'$ is the same as in the original code $\mathcal{C}$). The bits of the codeword in $\mathcal{C}'$ correspond to the *ends* of edges in the graph G (a uniform graph of degree D with N vertices has $DN/2$ edges, and each edge has 2 ends).

The transformation from $\mathcal{C}$ to $\mathcal{C}'$ is quite trivial: we place the bits of a codeword $\bar{x} = (x_1,\dots,x_N)$ from $\mathcal{C}$ to the corresponding vertices $(v_1,\dots,v_N)$ of G, and then rewrite the value of each bit x_i to the *opposite* ends of the edges (v_i,v_j) incident to v_i. In other words, for each edge (v_i,v_j) of the graph we assign the bits x_i and x_j from the codeword $\bar{x}$ to the opposite ends of this edge (x_i is assigned to the end incident to v_j, and x_j is incident to the end incident to v_i, see an example in Figure 1).

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

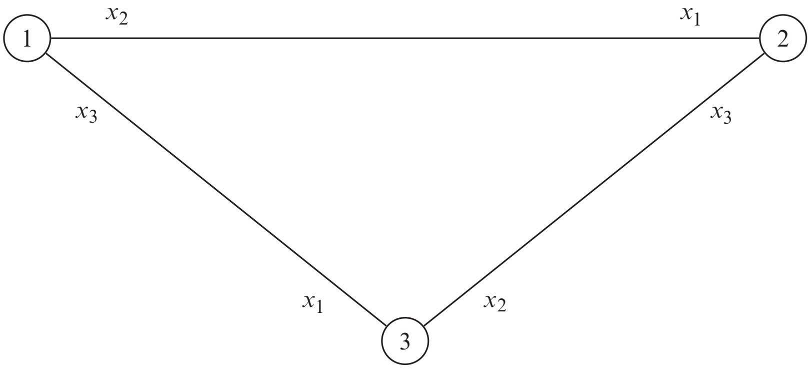

FIGURE 1. The bits $(x_1x_2x_3)$ assigned to the six ends of edges of the triangle graph.

Thus, we essentially duplicate D times each bit of a codewords and then redistribute these values between the ends of the edges of G.

The second stage of the amplification: from $\mathcal{C}'$ to $\mathcal{C}''$. We construct another code $\mathcal{C}''$, where each codeword consists of $N'' := 2^D N$ bits.

We remind that in the codewords of the code $\mathcal{C}'$ we have *two* bits assigned to every edge (one bit for each end of the edge), as shown in the example in Figure 1. To transform a codeword of $\mathcal{C}'$ in a codeword of $\mathcal{C}''$, we split the codeword in blocks of D bits corresponding to the *ends of edges* incident to one vertex. In other words, we group together the bits assigned to the edge's ends incident to one and the same vertex. (In the example in Figure 1, we obtain three blocks, with two bits in each block: (x_2,x_3), (x_1,x_3), and (x_1,x_2).) Then we separately encode each of these blocks by the Hadamard code[1].

Let us estimate the distance of $\mathcal{C}''$. The distance of the initial code $\mathcal{C}$ is equal to ρN, so every nonzero codeword $\bar{x}$ in this code contains at least ρN ones. If $\rho < \frac{1}{2} - \varepsilon$, then by strong ε-expansion, the corresponding codeword $\bar{x}'$ in $\mathcal{C}'$ contains $\geq (2+\varepsilon)\rho N$ blocks (with D bits in each block) that involve at least one nonzero bit. By applying the Hadamard code we obtain a codeword $\bar{x}''$ in $\mathcal{C}''$ with $\geq (2+\varepsilon)\rho N$ nonzero blocks of size 2^D, where each of these blocks contains exactly 2^{D-1} zeros and 2^{D-1} ones. Hence, the code distance of $\mathcal{C}''$ is not less than

$$\frac{(2+\varepsilon)\rho}{2} N'' = \left(1+\frac{\varepsilon}{2}\right)\rho N''.$$

Thus, we have "amplified" the relative code distance from ρ to $\left(1+\frac{\varepsilon}{2}\right)\rho$.

This described amplification procedure does not change the number of codewords in a code, and increases the code length by a constant factor. Hence, if the original family of codes $\mathcal{C}$ is asymptotically good, then the amplified codes $\mathcal{C}''$ are also asymptotically good.

[1] Instead of the Hadamard code (whose relative code distance is equal to $1/2$) we could take any other linear code with a relative code distance close enough to $1/2$.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

What is the structure of the checksum matrix of the new code? W.l.o.g. we may assume that the Hadamard code is systematic, so each bit of a codeword of $\mathcal{C}$ is just transferred to some position of the amplified $\mathcal{C}''$. Hence, for the amplified code we inherit all equations that define the original code $\mathcal{C}$ without any changes (up to renaming of the variables). If the checksum matrix of $\mathcal{C}$ is sparse, then each of these equations involve only $O(1)$ variables. Besides the equation inherited from $\mathcal{C}$, we add the equation that define the Hadamard code for each block of 2^D bits (one block for each vertex of G). Each of these equations contains at most $2^D = O(1)$ variables. Thus, the transformation preserves the property of sparsity of the checksum matrix. If the resulting family of linear constraints is not linearly independent, we can eliminate the redundant equations.

By repeating this amplification procedure $O(1)$ times we obtain a linear code with the relative distance $\geq \frac{1}{2} - \delta$. This code satisfies condition (L$'$), and therefore (L). The properties (G) and (S) are satisfied for this code, as explained above. It remains to achieve the property (M). We discuss this in the next section.

6.2.3. A checksum matrix with the mixing property. In this section we focus on the properties of a checksum matrix that look more natural being formulated as properties of a system of linear equations. Given a real $\sigma > 0$ and an integer $q > 0$, we study the following problem of linear algebra: we need to construct a system of $m \leq \sigma n$ linear equations with n variables, with the properties of sparsity (S) and mixing (M$'$). This problem can be reformulated as follows: we need to construct a bipartite graph $G = (V_L, V_R, E)$ with n vertices in the left part (the n vertices in V_L correspond to the variables of the system) and m vertices in the right part (the m vertices in V_R correspond to the linear equations of the system); an edge connects $v_i \in V_L$ with $w_j \in V_R$ (i.e., $(v_i, w_j) \in E$), if and only if the ith variable is involved in the jth equation of the linear system. In terms of this graph, the property of sparsity means that the degrees of all vertices in the right part are bounded by $O(1)$, and the property of mixing means that for every set S of βn vertices on the left there exists a set T of γm vertices on the right such that every $w \in T$ is connected with S by at least q edges.

We construct the required bipartite graph using an algebraic expander. Let us fix an algebraic expander H with parameters (N, d, λ). In our construction the vertices in V_R (the linear equations) will correspond to the vertices of this expander, and the vertices in V_L (the variables of the linear system) correspond to the paths of length t in the expander. Thus, the number of variables in the system is $n = Nd^t$, and the number of equations is $m = N$. The ith variable is involved in the jth equation, if and only if the ith path of length t in H contains the jth vertex of H. If t is a constant (does not depend on N), then this construction gives a sparse linear system: every variable is involved in $(t+1) \cdot d^t = O(1)$ equations. By choosing t large enough, we can make the fraction m/n less than the given parameter σ.

With a large enough t we obtain a linear system with the strong mixing property. Indeed, let us fix in H some set of vertices T of size γN. Then the fraction of paths P of length t where all but q vertices belong to T, is not greater than

$$(\gamma + \lambda)^{t-q} \cdot \binom{t}{q}$$

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

see [15, Theorem 3.10]. We can choose t and λ so that this fraction is less than β. Thus, we can achieve the property of strong mixing for the required values of β, γ, q.

6.2.4. Merging two constructions. Let us summarize the constructions discussed in two previous sections. In Section 6.2.2 we obtained a system of linear equations

$$\begin{cases} a_{1,1}x_1 + \cdots + a_{1,n}x_n & = & 0, \\ & \dots & \\ a_{m,1}x_1 + \cdots + a_{m,n}x_n & = & 0, \end{cases}$$

with n variables, with the dimension of the solution space $> c_1 n$ (property (G)), where every equation involves $\leq c_2$ variables (property (S)), for some positive constants c_1, c_2 that does not depend on n. In Section 6.2.3 we constructed another linear system

$$\begin{cases} b_{1,1}x_1 + \cdots + b_{1,n}x_n & = & 0, \\ & \dots & \\ b_{\ell,1}x_1 + \cdots + b_{\ell,n}x_n & = & 0, \end{cases}$$

with n variables and $\ell = \sigma n$ equations, with the property of strong mixing (M') from some β, γ, q. We are free to choose the parameters, so we may assume that $q > c_2$. Moreover, we can achieve the property (M') with arbitrarily small positive σ.

Now we joint together these two linear systems (we just take the union of linear equations from both systems). The resulting linear system is defined by the matrix

$$\begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & a_{2,3} & \dots & a_{2,n} \\ \dots & & & & \\ \dots & & & & \\ a_{m,1} & a_{m,2} & a_{m,3} & \dots & a_{m,n} \\ & & & & \\ b_{1,1} & b_{1,2} & b_{1,3} & \dots & b_{1,n} \\ \dots & & & & \\ b_{\ell,1} & b_{\ell,2} & b_{\ell,3} & \dots & b_{\ell,n} \end{pmatrix}.$$

This linear system defines a code which is still asymptotically good provided that $\sigma < c_1$ (we may choose the value of σ small enough). We refer to the m linear constraints that came to the joint system from the construction of Section 6.2.2 as *primary* equations, and to the ℓ linear constraints that came from the construction of Section 6.2.2 as *auxiliary* equations.

The new code has only a weak version of the mixing property: the property (M') holds only for the auxiliary equations. How to spread the property (M') to the entire linear system? To this end we xor the auxiliary equations with the primary equations. More precisely, we add (in the sense of linear algebra) each of the auxiliary equations to a fraction $1/\ell$ of the primary equations. The same time, the auxiliary equations

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

remain intact. Thus, we obtain a linear system with the matrix

$$\begin{pmatrix}
a_{1,1} \oplus b_{1,1} & a_{1,2} \oplus b_{1,2} & a_{1,3} \oplus b_{1,3} & \dots & a_{1,n} \oplus b_{1,n} \\
a_{2,1} \oplus b_{2,1} & a_{2,2} \oplus b_{2,2} & a_{2,3} \oplus b_{2,3} & \dots & a_{2,n} \oplus b_{2,n} \\
\dots & & & & \\
a_{\ell,1} \oplus b_{\ell,1} & a_{\ell,2} \oplus b_{\ell,2} & a_{\ell,3} \oplus b_{\ell,3} & \dots & a_{\ell,n} \oplus b_{\ell,n} \\
 & & & & \\
a_{\ell+1,1} \oplus b_{1,1} & a_{\ell+1,2} \oplus b_{1,2} & a_{\ell+1,3} \oplus b_{1,3} & \dots & a_{\ell+1,n} \oplus b_{1,n} \\
a_{\ell+2,1} \oplus b_{2,1} & a_{\ell+2,2} \oplus b_{2,2} & a_{\ell+2,3} \oplus b_{2,3} & \dots & a_{\ell+2,n} \oplus b_{2n} \\
\dots & & & & \\
a_{2\ell,1} \oplus b_{\ell,1} & a_{2\ell,2} \oplus b_{\ell,2} & a_{2\ell,3} \oplus b_{\ell,3} & \dots & a_{2\ell,n} \oplus b_{\ell,n} \\
 & & & & \\
a_{2\ell+1,1} \oplus b_{1,1} & a_{2\ell+1,2} \oplus b_{1,2} & a_{2\ell+1,3} \oplus b_{1,3} & \dots & a_{2\ell+1,n} \oplus b_{1,n} \\
a_{2\ell+2,1} \oplus b_{2,1} & a_{2\ell+2,2} \oplus b_{2,2} & a_{2\ell+2,3} \oplus b_{2,3} & \dots & a_{2\ell+2,n} \oplus b_{2n} \\
\dots & & & & \\
a_{3\ell,1} \oplus b_{\ell,1} & a_{3\ell,2} \oplus b_{\ell,2} & a_{3\ell,3} \oplus b_{\ell,3} & \dots & a_{3\ell,n} \oplus b_{\ell,n} \\
 & & & & \\
\dots & & & & \\
\dots & & & & \\
 & & & & \\
b_{1,1} & b_{1,2} & b_{1,3} & \dots & b_{1,n} \\
\dots & & & & \\
b_{\ell,1} & b_{\ell,2} & b_{\ell,3} & \dots & b_{\ell,n}
\end{pmatrix}.$$

This transform does not change the solution space of the system. The same time, the transformed system satisfies the mixing property (spread on all equations). Notice that when we xor an auxiliary equation with some of the primary equations, some 1's from both equations may cancel out. So we cannot claim that the modified system satisfies $(\mathbf{M}')$. However, since $q > c_2$, we can claim that the entire matrix satisfies the weaker property $(\mathbf{M})$.

If the equations of the resulting system are not linearly independent, we can eliminate the redundant equations. The number of eliminated equations is not greater than σn (the number of all auxiliary equations), and therefore this operation will not much affect the mixing property: in the worst case the value of parameter γ shrinks to $\gamma' = \gamma - \sigma$. Since the constructions in Sections 6.2.2–6.2.3 work for any $\gamma < 1$ and any $\sigma > 0$, we can obtain as the result any $\gamma' < 1$.

6.2.5. The algebraic expanders involved in the construction. In our construction we employed algebraic expanders with different parameters. In Section 6.2.2 we used a series of algebraic expanders with the parameters

$$(N_0, d, 1/2),\ (N_1 = N_0 \cdot 2^{D(\varepsilon)}, d, 1/2), \dots, (N_s = N_0 \cdot 2^{sD(\varepsilon)}, d, 1/2)$$

for some constants d and s. In Section 6.2.3 we used another algebraic expanders with the parameters (n_s, d, λ) for a small enough λ and a suitable d. The required algebraic expanders can be constructed explicitly, for example, by using the recursive schemes based on the zig-zag product, see [15]. This observation concludes the proof of Theorem 6.4.

§7. Conclusion. In the prolific paper [8], Cook and Reckhow proposed a very general definition of a *proof system*. The OBDD($\wedge$) and OBDD($\wedge$, reordering) studied in this paper are two specific examples of proof systems in the sense of Cook and Reckhow. Indeed, it is enough to observe that the following three properties are satisfied for these OBDDs:

1. There is a polynomial-time algorithm that verifies whether a function represented by an OBDD is satisfiable (see for example [29, Theorem 3.3.1]).
2. Let D_1, D_2, D_3 be some OBDDs in the same order. It is possible to check whether $D_1 = D_2 \wedge D_3$ in polynomial time (see for example [29, Theorem 3.3.6]).
3. There is a polynomial-time algorithm that verifies whether an OBDD D_1 in an order π_1 and an OBDDD_2 in an order π_2 represent the same function (see for example [29, Theorem 5.7.10]).

We believe that our techniques can be applied in a more general setting and extended to some other proof systems, with possibly other data structures representing the Boolean functions. More specifically, we argue that our techniques apply to the k-OBDDs. To outline this generalization, we survey our proofs and take inventory of the properties of the OBDD($\wedge$, reordering)s that were used in our argument. In our lower bounds we used the following property of the OBDDs:

4. If $f : \{0,1\}^n \to \{0,1\}$ is represented by a π-OBDD of size S, then for every partition (Π_0, Π_1) of $[n]$ such that elements of Π_0 precede elements of Π_1 in the order π, the communication complexity of f with respect to (Π_0, Π_1) is $O(\log S)$.

This is essentially the only property of OBDD($\wedge$, reordering) used in Section 3. Indeed, we prove lower bounds for the complexity of OBDD($\wedge$, reordering) with the following argument:

- In the first step, we prove exponential lower bounds on the OBDD complexity of a satisfiable formula Φ. Actually, in each case we prove a linear lower bound on the communication complexity of computing Φ with respect to every nearly balanced partition of the input.
- We focus on the last step of the refutation. This step is the conjunction of F_1 and F_2 in the same order π. We need to prove that at least one of F_1 and F_2 has exponential size. Both F_1 and F_2 are satisfiable, and they are conjunctions of different sets of clauses of the initial formula. We construct partial substitutions ρ_1 and ρ_2 with the same support such that the formula $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is isomorphic to Φ. Then the communication complexity of computing $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is linear with respect to any nearly balanced partition Π consistent with π. Hence, the communication complexity of $F_1|_{\rho_1}$ or $F_2|_{\rho_2}$ with respect to Π is linear. Thus, the communication complexity of either F_1 or F_2 with respect to π is linear. Therefore, by Property 4 of OBDDs it follows that for the order π either F_1 or F_2 has exponential size.

Thus, only Property 4 is needed to prove a lower bound for OBDDs. It is known that k-OBDDs for $k = O(1)$ enjoy Properties 1-3 and Property 4 as well (see for example [29]). (A binary decision diagram is called k-OBDD if every path from the source to a leaf can be split into at most k parts such that in each part the variables appears

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

with respect to a fixed order π.) Therefore, our lower bounds proven in Section 3, equally apply to the proof systems based on k-OBDDs. However, it is not known whether these k-OBDD proof systems are stronger than OBDD proof systems.

In the paper we implement this scheme to prove lower bounds on Tseitin formulas and the pigeonhole principle. The pigeonhole principle and Tseitin formulas on biconnected graphs are minimally unsatisfiable; we used this property to prove that $F_1|_{\rho_1} \wedge F_2|_{\rho_2}$ is isomorphic to a hard satisfiable formula Φ. However, this method can be applied to not minimally unsatisfiable formulas provided that the formula has a simple structure of minimally unsatisfiable subformulas. For example, the proof of the lower bound on the pigeonhole principle can be almost directly translated into a proof of a lower bound on the functional onto pigeonhole principle.

In Section 4, we constructed a family of formulas that have short OBDD($\wedge$, reordering) refutations but all the OBDD($\wedge$) refutations of them have exponential size. And again, to get the lower bound, we used only Property 4 of OBDDs. So this lower bound holds for k-OBDD($\wedge$) as well.

Finally, in Section 5, we proved that while an OBDD($\wedge, \exists$, reordering) algorithm processes a formula, at some moment it comes to a diagram D that computes a function with a linear communication complexity. By Property 4, this implies an exponential lower bound on the size of D. Therefore, the proven lower bound holds also for the k-OBDD($\wedge, \exists$, reordering) algorithms.

Further research. The major open problem is to prove superpolynomial lower bounds for OBDD($\wedge$, weakening, reordering) refutations.

Recently Buss et al. [5] established an exponential separation between OBDD($\wedge$, weakening, reordering) and OBDD($\wedge$, weakening) proof systems. This result and Theorem 4.8, showing that reordering rule is useful, are based on quite artificial formulas; thus it is interesting to find separating CNF formulas with clear meanings. Tseitin formulas based on expanders are easy for OBDD($\wedge$, weakening), but by Theorem 3.15 they are hard for OBDD($\wedge$, reordering). The paper [5] presented a family of formulas such that their shortest OBDD($\wedge$, weakening) refutation is *superpolynomially* larger than every OBDD($\wedge$, reordering) refutation of these formulas; it would be interesting to strengthen this result and establish an *exponential* separation.

Acknowledgments. The authors are grateful to Sam Buss for fruitful discussions and to the anonymous reviewers for useful comments.

The research presented in Sections 3, 4, and 5 was supported by Russian Science Foundation (project 16-11-10123). The work of the third author on the research presented in Section 6 was partially supported by RFBR grant 19-01-00563.

Dmitry Itsykson is a Young Russian Mathematics award winner and would like to thank sponsors and jury of the contest.

REFERENCES

[1] A. S. M. Aguirre and M. Y. Vardi, *Random 3-sat and bdds: The plot thickens further*, ***Principles and Practice of Constraint Programming—CP 2001*** (T. Walsh, editor), Lecture Notes in Computer Science, vol. 2239, Springer, Berlin, 2001, pp. 121–136.

[2] N. Alon and F. R. K. Chung, *Explicit construction of linear sized tolerant networks*. ***Discrete Mathematics***, vol. 306 (2006), no. 10–11, pp. 1068–1071.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

[3] A. Atserias, P. G. Kolaitis, and M. Y. Vardi, *Constraint propagation as a proof system*, ***Principles and Practice of Constraint Programming—CP 2004*** (M. Wallace, editor), Lecture Notes in Computer Science, vol. 3258, Springer, Berlin, 2004, pp. 77–91.

[4] M. L. Bonet, T. Pitassi, and R. Raz, *Lower bounds for cutting planes proofs with small coefficients*, this Journal, vol. 62 (1997), no. 3, pp. 708–728.

[5] S. Buss, D. Itsykson, A. Knop, and D. Sokolov, *Reordering rule makes OBDD proof systems stronger*, ***33rd Computational Complexity Conference, CCC 2018*** (R. A. Servedio, editor), Leibniz International Proceedings in Informatics, Schloss Dagstuhl, Dagstuhl, 2018, pp. 16:1–16:24.

[6] J. Cheeger, *A lower bound for the smallest eigenvalue of the Laplacian*. ***Problems in Analysis***, vol.78 (1970), p. 195.

[7] W. Chén and W. Zhang, *A direct construction of polynomial-size OBDD proof of pigeon hole problem*. ***Information Processing Letters***, vol. 109 (2009), no. 10, pp. 472–477.

[8] S. A. Cook and R. A. Reckhow, *The relative efficiency of propositional proof systems*, this Journal, vol. 44 (1979), no. 1, pp. 36–50.

[9] W. J. Cook, C. R. Coullard, and G. Turán, *On the complexity of cutting-plane proofs*. ***Discrete Applied Mathematics***, vol. 18 (1987), no. 1, pp. 25–38.

[10] P. Duris, J. Hromkovic, S. Jukna, M. Sauerhoff, and G. Schnitger, *On multi-partition communication complexity*. ***Information and Computation***, vol. 194 (2004), no. 1, pp. 49–75.

[11] L. Friedman and Y. Xu, *Exponential lower bounds for refuting random formulas using ordered binary decision diagrams*, ***Computer Science—Theory and Applications*** (A. A. Bulatov and A. M. Shur, editors), Lecture Notes in Computer Science, vol. 7913, Springer, Berlin, 2013, pp. 127–138.

[12] R. G. Gallager, *Low-density parity-check codes*. ***Institute of Electrical and Electronics Engineers. Transactions on Information Theory***, vol. 8 (1962), no. 1, pp. 21–28.

[13] J. F. Groote and H. Zantema, *Resolution and binary decision diagrams cannot simulate each other polynomially*. ***Discrete Applied Mathematics***, vol. 130 (2003), no. 2, pp. 157–171.

[14] V. Guruswami, *List decoding from erasures: bounds and code constructions*. ***Institute of Electrical and Electronics Engineers. Transactions on Information Theory***, vol. 49 (2003), no. 11, pp. 2826–2833.

[15] S. Hoory, N. Linial, and A. Wigderson, *Expander graphs and their applications*. ***Bulletin of the American Mathematical Society***, vol. 43 (2006), no. 4, pp. 439–561.

[16] R. Impagliazzo, T. Pitassi, and A. Urquhart, *Upper and lower bounds for tree-like cutting planes proofs*, ***Proceedings of the Ninth Annual IEEE Symposium on Logic in Computer Science***, Institute of Electrical and Electronics Engineers, Piscataway, NJ, 1994, pp. 220–228.

[17] D. Itsykson, A. Knop, A. E. Romashchenko, and D. Sokolov, *On OBDD-based algorithms and proof systems that dynamically change order of variables*, ***34th Symposium on Theoretical Aspects of Computer Science*** (H. Vollmer and B. Vallée, editors), Leibniz International Proceedings in Informatics, Schloss Dagstuhl, Dagstuhl, 2017, pp. 43:1–43:14.

[18] J. Krajícek, *Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic*, this Journal, vol. 62 (1997), no. 2, pp. 457–486.

[19] ———, *An exponential lower bound for a constraint propagation proof system based on ordered binary decision diagrams*, this Journal, vol. 73 (2008), no. 1, pp. 227–237.

[20] E. Kushilevitz and N. Nisan, ***Communication Complexity***, Cambridge University Press, Cambridge, 1997.

[21] A. Lubotzky, R. Phillips, and P. Sarnak, *Ramanujan graphs*. ***Combinatorica***, vol. 8 (1988), no.3, pp. 261–277.

[22] C. Meinel and A. Slobodova, *On the complexity of constructing optimal ordered binary decision diagrams*. ***Proceedings of Mathematical Foundations of Computer Science***, vol. 841 (1994), pp. 515–524.

[23] C. Meinel and T. Theobald, ***Algorithms and Data Structures in VLSI Design: OBDD—Foundations and Applications***, Springer, New York, 1998.

[24] G. Pan and M. Y. Vardi, *Symbolic techniques in satisfiability solving*. ***Journal of Automated Reasoning***, vol. 35 (2005), no. 1–3, pp. 25–50.

[25] N. Segerlind, *Nearly-exponential size lower bounds for symbolic quantifier elimination algorithms and OBDD-based proofs of unsatisfiability*, preprint, 2007, arXiv:cs/0701054.

[26] ———, *On the relative efficiency of resolution-like proofs and ordered binary decision diagram proofs*. ***Electronic Colloquium on Computation Complexity***, vol. 126 (2007), p. 23.

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press

[27] O. Tveretina, C. Sinz, and H. Zantema, An exponential lower bound on OBDD refutations for pigeonhole formulas. ***Proceedings Fourth Athens Colloquium on Algorithms and Complexity*** (E. Markakis and I. Milis, editors), Electronic Proceedings in Theoretical Computer Science, vol. 4, pp. 13–21.

[28] A. Urquhart, *Hard examples for resolution*. ***Journal of the ACM***, vol. 34 (1987), no. 1, pp. 209–219.

[29] I. Wegener, ***Branching Programs and Binary Decision Diagrams***, SIAM, New York, 2000.

[30] G. Zémor, *On expander codes*. ***IEEE Transactions on Information Theory***, vol. 47 (2001), no. 2, pp. 835–837.

ST. PETERSBURG DEPARTMENT OF V.A.
STEKLOV INSTITUTE OF MATHEMATICS
RUSSIAN ACADEMY OF SCIENCE
ST. PETERSBURG, RUSSIA
E-mail: dmitrits@pdmi.ras.ru

ST. PETERSBURG DEPARTMENT OF V.A.
STEKLOV INSTITUTE OF MATHEMATICS
RUSSIAN ACADEMY OF SCIENCE
ST. PETERSBURG, RUSSIA
E-mail: aknop@ucsd.edu

LIRMM
UNIV MONTPELLIER, CNRS
MONTPELLIER, FRANCE
and
ON LEAVE FROM IITP RAS
MOSCOW, RUSSIA
E-mail: andrei.romashchenko@lirmm.fr

ST. PETERSBURG DEPARTMENT OF V.A.
STEKLOV INSTITUTE OF MATHEMATICS
RUSSIAN ACADEMY OF SCIENCE
ST. PETERSBURG, RUSSIA
E-mail: sokolov.dmt@gmail.com

https://doi.org/10.1017/jsl.2019.53 Published online by Cambridge University Press