

# Lower bounds for myopic DPLL algorithms with a cut heuristic \*

Dmitry Itsykson and Dmitry Sokolov

March 26, 2012

## Abstract

The paper is devoted to lower bounds on the time complexity of DPLL algorithms that solve the satisfiability problem using a splitting strategy. Exponential lower bounds on the running time of DPLL algorithms on unsatisfiable formulas follow from the lower bounds for resolution proofs. Lower bounds on satisfiable instances are also known for some classes of DPLL algorithms; this lower bounds are usually based on reductions to unsatisfiable instances. In this paper we consider DPLL algorithms with a cut heuristic that may decide that some branch of the splitting tree will not be investigated. DPLL algorithms with a cut heuristic always return correct answer on unsatisfiable formulas while they may err on satisfiable instances. We prove the theorem about effectiveness vs. correctness trade-off for deterministic myopic DPLL algorithms with cut heuristic. Myopic algorithms can see formulas with erased signs of negations; they may also request a small number of clauses to read them precisely.

We construct a family of unsatisfiable formulas  $\Phi^{(n)}$  and for every deterministic myopic algorithm  $A$  we construct polynomial time samplable ensemble of distributions  $R_n$  concentrated on satisfiable formulas such that if  $A$  gives a correct answer with probability 0.01 on a random formula from the ensemble  $R_n$  then  $A$  runs exponential time on the formulas  $\Phi^{(n)}$ .

One of the consequences of our result is the existence of polynomial time samplable ensemble of distributions  $Q_n$  concentrated on satisfiable formulas such that every deterministic myopic algorithm that gives a correct answer with probability  $1 - o(1)$  on a random formula from the ensemble  $Q_n$  runs exponential time on the formulas  $\Phi^{(n)}$ .

Other consequence is the following statement: for every deterministic myopic algorithm  $A$  we construct a family of satisfiable formulas  $\Psi^{(n)}$  and polynomial time samplable ensemble of distributions  $R_n$  concentrated on satisfiable formulas such that if  $A$  gives a correct answer with probability 0.01 on a random formula from the ensemble  $R_n$  then  $A$  runs exponential time on the formulas  $\Psi^{(n)}$ .

Keywords: DPLL, lower bounds, heuristic computation, expander

---

\*The work is partially supported by RAS Program for Fundamental Research, the president grants NSh-3229.2012.1 and MK-4108.2012.1, by RFBR grants 11-01-00760-a and 11-01-12135-ofi-m-2011, by Rokhlin's scholarship and by PDMI Computer Science Club scholarship.

# 1 Introduction

DPLL (are named by the authors: Davis, Putnam, Logemann and Loveland) algorithms are one of the most popular approach to the problem of satisfiability of Boolean formulas (SAT). DPLL algorithm is a recursive algorithm that takes the input formula  $\phi$ , uses a procedure **A** to choose a variable  $x$ , uses a procedure **B** that chooses the value  $a \in \{0, 1\}$  for the variable  $x$  that would be investigated first, and makes two recursive calls on inputs  $\phi[x := a]$  the  $\phi[x := 1 - a]$ . Note that the second call is not necessary if the first one returns the result, that the formula is satisfiable.

There is a number of works concerning lower bounds for DPLL algorithms: for unsatisfiable formulas exponential lower bounds follow from lower bounds on the complexity of resolution proofs [Urq87], [Tse68]. In case of satisfiable formulas we have no hope to prove superpolynomial lower bound since if  $P = NP$ , then procedure **B** may always choose the correct value of the variable according to some satisfying assignment. The paper [AHI05] gives exponential lower bounds for two wide enough classes of DPLL algorithms: myopic and drunken algorithms. In the myopic case procedures **A** and **B** can see formula with erased signs of negation, they can request the number of positive and negative occurrences for every variable and also may read  $K = n^{1-\epsilon}$  clauses precisely. In the drunken algorithms the procedure **A** may be arbitrary while the procedure **B** chooses the value uniformly at random. The paper [CEMT09] shows that myopic algorithms invert Goldreich's function ([Gol00]) based on a random graph in at least exponential time, [Its10] and [Mil09] extend this result for drunken algorithms. The paper [IS11] presents the explicit Goldreich's function based on any expander that is hard to invert for drunken and myopic algorithms.

All discussed lower bounds for satisfiable instances are based on the fact that during several first steps algorithm falls into a hard unsatisfiable formula, and algorithm should investigate the whole it' splitting tree. In this work we extend the class of DPLL algorithms by adding the procedure **C** that may decide that some branch of the splitting tree will not be investigated since it is not too "perspective". More precisely, before each recursive call an algorithm calls the procedure **C** that decides whether to make this recursive call or not. DPLL algorithms with cut heuristic always give a correct answer on unsatisfiable formulas; however they may err on satisfiable formulas. On the other hand if the presence of a cut heuristic gives the substantial improvement on the time complexity while the bad instances (i.e. instances on which the algorithm errs) are not easy to find, then such algorithms become reasonable.

In this work we show that it is possible to construct the family of unsatisfiable formulas  $\Phi^{(n)}$  in polynomial time such that for every myopic deterministic heuristics **A** and **C** there exists a polynomial time samplable ensemble of distributions  $R_n$  such that the DPLL algorithm based on procedures **A**, **B** and **C** for some **B** either errs on 99% of random inputs according  $R_n$  or runs exponential time on formulas  $\Phi^{(n)}$ . In case **A** and **C** are not restricted we show that a statement similar to above is equivalent to  $P \neq NP$ . The case of randomized myopic procedures **A** and **C** is left open.

In the proof of the main result we use the technique of closures for expander graphs invented by Alekhovich. We suggest a constructive variant of the closure notion and apply it to the construction of the ensemble of distributions  $R_n$ .

We also give similar lower bound on satisfiable instances but in case of satisfiable for-

mulas it is impossible to prove precisely the same statement since myopic algorithm may be trained for this particular formula  $\Phi^{(n)}$ . Therefore in satisfiable case the construction of formula  $\Phi^{(n)}$  should be depended on procedures **A**, **B**. And we also assume that **B** is myopic.

The constructed distribution  $R_n$  depends on procedures **A** and **C**. We also construct a universal ensemble of distributions  $Q_n$  that dominates all possible ensembles  $R_n$ . In particular we prove that if myopic DPLL with a cut heuristic errs with  $o(1)$  probability on a random formula from the ensemble  $Q_n$ , then it runs exponential time on formulas  $\Phi^{(n)}$ .

**Heuristic acceptors.** The study of DPLL algorithms with cut heuristic was also motivated by the study of heuristic acceptors [HIMS11]. The distributional proving problem is a pair  $(L, D)$  of a language  $L$  and a polynomial time samplable distribution  $D$  concentrated on the complement of  $L$ . An algorithm  $A$  is called a heuristic acceptor if it has additional input  $d$  that represents the parameter of the error and for every  $x \in L$  and  $d \in \mathbb{N}$ ,  $A(x, d)$  returns 1 and  $\Pr_{x \leftarrow D_n}[A(x) = 1] < 1/d$  for every integer  $n$ . We call an acceptor polynomially bounded if for every  $x \in L$  running time of  $A(x, d)$  is bounded by polynomial in  $|x| \cdot d$ . The paper [HIMS11] shows that the existence of distributed proving problems that have no polynomially bounded acceptors is equivalent to the existence of infinitely often one-way functions.

Let  $D$  be some distribution concentrated on satisfiable formulas. We consider DPLL algorithm with a cut heuristic supplied with an additional parameter  $d$  that is available for procedures **A**, **B**, **C**. We call such an algorithm a heuristic DPLL acceptor if it satisfies the definition of a heuristic acceptor. Our result implies that there are no deterministic polynomially bounded myopic DPLL acceptors for the proving problem  $(UNSAT, Q)$ .

**Priority branching tree.** Priority branching trees were introduced in the paper [ABBO<sup>+</sup>09] as computational model that extends ideas of dynamic programming and backtracking algorithms. We briefly describe it with application to SAT. A formula is represented as set of data items. Every data item consists of a variable and all clauses that contain this variable. The formula is described by the set of all data items:  $\mathcal{D} = \{D_1, D_2, \dots, D_n\}$ . Branching tree is determined by two family of functions  $r_k : \mathcal{D}^k \times \{0, 1\}^k \rightarrow \mathcal{O}(\mathcal{D})$  and  $c_k : \mathcal{D}^{k+1} \times \{0, 1\}^k \rightarrow \mathcal{O}(\{0, 1, \perp\})$ , where  $\mathcal{O}(S)$  is set of all orders on the set  $S$ . Every vertex of branching tree is marked by a data item from  $\mathcal{D}$  and branching is based on the value of variable from the current data item. The data item in the current vertex is chosen according function  $r_k$  that gets on the input all data items and values of variables on the path from the root to current vertex. Function  $r_k$  returns the order on data items and the current data item equals the first unused data item according the order. Function  $c_k$  determines the order in which values of a variable are investigated. Tree includes branches for values that precedes  $\perp$  in the order. So the priority branching tree may be incomplete and some solutions may be lost.

The paper [ABBO<sup>+</sup>09] gives exponential lower bound on the depth first size of the priority branching tree on formulas that code a system of linear equalities modula 2 under assumption that the algorithm correctly solves all problems of that type. In this paper (as it was in [ABBO<sup>+</sup>09]) we consider formulas of the type  $Ax = b$  with expander based matrix  $A$  such that the number of nonzero elements on every row is bounded by

a constant. Running of a priority branching tree on such formulas is actually a myopic algorithm with cut heuristic. If functions  $r_k$  and  $c_k$  are polynomially bounded our result may be extent to correctness vs effictiveness tradeoff for priority branching trees.

**Related conditional result.** The paper [GSTS07] gives the following nice result: if  $P \neq NP$ , then for every polynomial time SAT algorithm  $A$  there exists a polynomial time generator of hard instances  $G$ . I.e. for almost all  $n$  the algorithm  $G(1^n)$  returns a satisfiable formula such that  $A$  is failed to find it's satisfying assignment. Our result is unconditional but it may be applied to very restricted class of algorithms. We also note that our lower bound is not just superpolynomial but it is subexponential, while the distribution of hard instances is samplable in polynomial time.

## 2 Preliminaries

By partial substitution we mean the set of assignments  $x_i := a_i$ , where  $x_i$  is a propositional variable and  $a_i \in \{0, 1\}$  such that every propositional variable has at most one occurrence in it. For partial substitution  $\rho$  we denote  $vars(\rho)$  the set of variable that appears in  $\rho$ . For propositional formula  $\phi$  we denote by  $\phi|_\rho$  the formula that is obtained from  $\phi$  after applying all assignments from  $\rho$  and elementary simplifications.

An *ensemble of distributions* is a family  $D_n$ , where  $D_n$  is a distribution on the finite set of strings. The ensemble of distributions  $D_n$  is polynomial time samplable if there is a polynomial time randomized algorithm  $S$  (a sampler) such that for every  $n$  outputs of  $S(1^n)$  are distributed according to  $D_n$ .

### 2.1 DPLL algorithms with a cut heuristic

We define a family of algorithms that solve the satisfiability problem of CNF formulas. DPLL algorithms with a cut heuristic are parameterized by three procedures: **A**, **B** and **C**. The procedure **A** takes a formula as the input and returns a variable for splitting; the procedure **B** takes a formula and a variable as the input and returns a value that would be invstigated first; the procedure **C** takes a formula, variable and its value and decides whether an algorithm should investigate this branch.

Formally algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  is defined by the following way.

**Algorithm 2.1.** Algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$ .

- Input: formula  $\varphi$
- If  $\varphi$  contains no clauses, return 1.
- If  $\varphi$  contains empty clause, return 0.
- $x := \mathbf{A}(\varphi)$ ;
- $b := \mathbf{B}(\varphi, x)$ ;
- If  $\mathbf{C}(\varphi, x, b) = 1$ , then if  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\varphi[x := b]) = 1$ , return 1.

- If  $\mathbf{C}(\varphi, x, \neg b) = 1$ , then if  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\varphi[x := \neg b]) = 1$ , return 1.
- Return 0.

Let  $\mathbf{1}$  denote the constant 1 function. Algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$  obviously correctly solves the satisfiability problem. For every procedures  $\mathbf{A}, \mathbf{B}, \mathbf{C}$  the algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  returns the correct answer on all unsatisfiable formulas, however its answer on satisfiable formulas may be incorrect.

For every formula  $\varphi$  execution of an algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  corresponds to a splitting tree. Vertices of a splitting tree correspond to recursive calls, edges correspond to substitutions of a value to a variable. Every leaf of a splitting tree corresponds to one of the three possible situations: 1) some clause of initial formula is refuted; 2) the substitution is a satisfying assignment; 3) the procedure  $\mathbf{C}$  reports 0 two times. It is not hard to see that if procedures  $\mathbf{A}$  and  $\mathbf{B}$  are deterministic, then the splitting tree of  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  is a subtree of the splitting tree of  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}$ ; if the input formula  $\varphi$  is unsatisfiable, then the above statement holds even if only  $\mathbf{A}$  is deterministic.

By running time of the algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  we mean the number of recursive calls that equals to the number of vertices in the splitting tree.

Our main goal is to prove lower bounds for almost correct DPLL algorithms with a cut heuristic. Namely we construct the polynomial time samplable ensemble of distributions  $R_n$  concentrated on satisfiable formulas and the sequence of unsatisfiable formulas  $\Phi^{(n)}$  such that if  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  with high probability correctly solves the satisfiability problem for instances distributed according to  $R_n$ , then the algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  will run exponential time on formulas  $\Phi^{(n)}$ . Without restrictions on procedures  $\mathbf{A}, \mathbf{B}, \mathbf{C}$  it is unlikely that we prove the above statement since if  $P = NP$ , the polynomial time procedure  $\mathbf{C}$  may cut all unsatisfiable branches. Therefore we need to restrict the power of heuristic  $\mathbf{C}$  and we require it to be myopic.

A myopic procedure has access to the formula with erased signs of negations. For every variable it also has access to the number of its positive and negative occurrences. A myopic procedure is also able to read  $K = n^{1-\varepsilon}$  clauses of the formula precisely (with all negations).

However it is not enough to restrict only the procedure  $\mathbf{C}$  since the procedure  $\mathbf{A}$  may transmit information to the procedure  $\mathbf{C}$ ; for example  $\mathbf{A}$  may return lexicographically first variable for satisfiable formulas and lexicographically last value for unsatisfiable formulas and the procedure  $\mathbf{C}$  will just cut unsatisfiable branches. Therefore we restrict  $\mathbf{A}$  to be myopic. We also require for  $\mathbf{A}$  and  $\mathbf{C}$  to be deterministic. The case of randomized myopic  $\mathbf{A}$  and  $\mathbf{C}$  is left open.

### 3 Distribution

Here and after we assume that procedures  $\mathbf{A}$  and  $\mathbf{C}$  are deterministic and polynomial time myopic algorithms.

Let  $\Phi$  be an unsatisfiable formula and  $T_\Phi$  be the splitting tree of  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{1}}(\Phi)$ . We say that the set of vertices  $\{v_1, v_2, \dots, v_S\}$  of the tree  $T_\Phi$  is *the system of myopic copies* of  $\Phi$ , if the following is satisfied:

- for every  $i, j \in \{1, \dots, S\}$  a vertex  $v_i$  is not a descendant of  $v_j$ ;

- For every vertex  $v_i$  there exists a satisfiable formula  $\Phi_i$  with the only satisfying assignment such that the substitution made on the path from the root of  $T_\Phi$  to  $v_i$  is consistent with the satisfying assignment of  $\Phi_i$ .
- Myopic algorithms **A** and **C** can't distinguish formulas  $\Phi_i$  and  $\Phi$  on the path from the root of  $T_\Phi$  to  $v_i$ . Formally it means that for every substitution  $\rho$  that corresponds to this path formulas  $\Phi_i|_\rho$  and  $\Phi|_\rho$  with erased negation signs have no differences; every variable has the same number of positive and negative occurrences and sets of clauses that **A** and **C** read with negations are the same for formulas  $\Phi_i|_\rho$  and  $\Phi|_\rho$ .

We call formulas  $\{\Phi_1, \Phi_2, \dots, \Phi_S\}$  *myopic representatives* of  $\Phi$ .

**Lemma 3.1.** Let  $\Phi$  be an unsatisfiable formula and  $\{v_1, v_2, \dots, v_S\}$  is the system of myopic copies of  $\Phi$  with myopic representatives  $\{\Phi_1, \Phi_2, \dots, \Phi_S\}$ . Let  $U_\Phi$  be a uniform distribution on the set of formulas  $\{\Phi_1, \Phi_2, \dots, \Phi_S\}$ . Suppose that  $\Pr_{\phi \leftarrow U_\Phi}[\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$ , then the running time of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}(\Phi)$  is at least  $(1 - \epsilon)S$ . (The probability also includes random bits of the algorithm **B**).

*Proof.* Note that the usage of random bits by **B** does not affect the running time of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  on unsatisfiable formulas. We fix the sequence of random bits in such a way that if **B** uses this sequence then  $\Pr_{\phi \leftarrow U_\Phi}[\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$ . Since now we assume that **B** uses only this sequence of random bits.

Lets consider one of the formulas  $\Phi_j$  such that  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}(\Phi_j) = 1$ . We note that the splitting tree of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  on the input  $\Phi_j$  contains the path from the root to  $v_j$  from the tree  $T_\Phi$ . Since the only satisfying assignment of  $\Phi_j$  is consistent with the substitution made by this path, if  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  diverges from this path it will necessary come back since  $\Phi_j$  has the unique satisfying assignment. During this path the procedure **A** will select the same variables for splitting since it can't differ  $\Phi$  from  $\Phi_j$ . The procedure **C** also can't differ  $\Phi$  from  $\Phi_j$  during this path, therefore the vertex  $v_j$  is necessary in the splitting tree of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  on the input  $\Phi$ . Thus at least  $(1 - \epsilon)S$  vertices from the set  $\{v_1, \dots, v_S\}$  must be visited by  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  on the input  $\Phi$ .  $\square$

So our goal is the construction of the formula  $\Phi$  that has the system of myopic copies of exponential size. And moreover the uniform distribution on the myopic representatives of  $\Phi$  should be polynomial time samplable.

We consider formulas that code a system of linear equations  $Ax = b$  over the field  $\mathbb{F}_2$ , where  $A$  is a  $n \times n$  matrix.

We require that in every row of the matrix  $A$  the number of ones is bounded by the constant  $d$ . In this case the linear system consists of equations of type  $x_{i_1} + x_{i_2} + \dots + x_{i_s} = b_i$ , where  $s \leq d$ . Such equation may be written in  $d$ -CNF form with at most  $2^d$  clauses. Note that the  $d$ -CNF form of equalities with right hand sides 0 and 1 differs only in the literal signs and also the number of positive and negative occurrences for every variable is the same. Therefore myopic procedure can't see a bit of the right hand side until it requests at least one of corresponding clauses. In what follows we say that a myopic procedure opens a bit of the vector  $b$  if it requests a clause that correspondes to this bit.

We will choose a nonsingular matrix  $A$ , in this case the system  $Ax = b$  has a unique solution. It means that the variable  $x_1$  can't take some value  $\alpha \in \{0, 1\}$ . The formula  $\Phi$

denotes the formula that encodes the system  $Ax = b$  after the substitution  $x_1 := \alpha$ .  $\Phi$  is unsatisfiable by the choice of  $\alpha$ .

Now we may describe some intuition how to construct the system of myopic representatives of the formula  $\Phi$ . For vertices  $v_i$  from the splitting tree of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, 1}$  we have to find  $b' \in \{0, 1\}^n$  such that the solution of the system  $Ax = b'$  would be consistent with substitutions made during the path from the root to  $v_i$  and with  $x_1 := \alpha$ . However it is not clear why it is possible. We will do it by the appropriate choice of matrix  $A$ .

## 4 Boundary expanders and the closure

We consider a bipartite graph  $G$  (multiple edges are allowed), we denote its parts by  $X$  and  $Y$ . The boundary of the set  $I \subset Y$  is a set  $\delta(I) = \{x \in X \mid \text{there is exactly one edge between } I \text{ and } x\}$ . The graph  $G$  is called  $(r, d, c)$ -boundary expander if the degrees of all vertices from  $Y$  do not exceed  $d$  and for every set  $I \subseteq Y$  if  $|I| \leq r$ , then  $|\delta(I)| \geq c|I|$ .

Let  $G$  be an  $(r, d, c)$ -boundary expander. Let's  $J \subseteq X$ ; a closure of the set  $J$  is the maximum subset  $I \subseteq Y$  that satisfies the following properties: 1)  $|I| \leq r$ ; 2)  $\delta(I) \subseteq J$ . The closure may be not uniquely defined; we denote lexicographically first closure of the set  $J$  by  $Cl(J)$ .

**Proposition 4.1.** Let  $G$  be an  $(r, d, c)$ -boundary expander with  $c \geq 1$ , then the following properties hold. 1) If  $I$  is a closure of the set  $J$ , then  $|I| \leq \frac{|J|}{c}$ . 2) If  $|J| < r/2$ , there is the unique closure of the set  $J$ . 3) Let's  $J \subseteq J'$  and  $|J'| < r/2$ , then  $Cl(J) \subseteq Cl(J')$ .

*Proof.* 1) Since  $|I| \leq r$  and  $\delta(I) \subseteq J$ , we get  $|J| \geq \delta(I) \geq c|I|$ . 2) Let  $I_1, I_2$  be two closures of the set  $J$ . By item 1 of the Proposition we get that  $|I_j| \leq \frac{|J|}{c} < r/2$ . It is not hard to see that  $\delta(I_1 \cup I_2) \subseteq \delta(I_1) \cup \delta(I_2) \subseteq J$  and  $|I_1 \cup I_2| \leq r$ . From the maximality of a closure we get  $I_1 = I_2$ . 3) Since  $|J| \leq |J'| < r/2$ , there are unique closures for  $J$  and  $J'$  by item 2 of the Proposition. Let's consider the set  $Cl(J) \cup Cl(J')$ , obviously  $\delta(Cl(J) \cup Cl(J')) \subseteq J'$ , also item 1 of the Proposition implies  $|Cl(J) \cup Cl(J')| \leq 2|J'|/c < r$ . Therefore  $Cl(J') = Cl(J) \cup Cl(J')$ .  $\square$

For every bipartite graph  $G$  with parts  $X$  and  $Y$  we define a  $|Y| \times |X|$  matrix  $A$ . Elements of  $X$  correspond to columns, elements of  $Y$  correspond to rows.  $A_{y,x}$  is the number of edges between  $x$  and  $y$  modulo 2. Let  $G$  be an  $(r, d, c)$ -boundary expander with  $c \geq 1$ . We call  $A$  an adjacency matrix of  $G$ . We consider a linear system  $Ax = b$ , where  $x$  is a vector of unknowns of size  $|X|$ . Let  $\rho$  be a partial substitution for variables of  $x$ . A partial substitution  $\rho$  is called *locally consistent* if  $|\rho| < r/2$  and the system of equalities  $Ax|_{Cl(vars(\rho))} = b|_{Cl(vars(\rho))}$  has a solution that is consistent with  $\rho$ . Here and after for vector  $z$  by  $z|_E$  we denote the projection of  $z$  to the coordinates from a set  $E$ .

**Lemma 4.1** ([AHI05]). If a partial substitution  $\rho$  is locally consistent, then for every  $I \subseteq Y, |I| \leq r/2$  the system  $Ax|_I = b|_I$  has a solution that is consistent with  $\rho$ .

*Proof.* Let's consider the minimal set  $I, |I| \leq r/2$  that does not satisfy the statement of the Lemma. Assume that  $\delta(I) \not\subseteq vars(\rho)$ , let's  $y \in I$  such that for some vertex  $z \in X \setminus vars(\rho)$  there is exactly one edge from  $z$  to  $I$  and it connects  $z$  with  $y$ . From the minimality of  $I$  we get that  $I' = I \setminus \{y\}$  satisfies the statement of the Lemma; i.e.

there is  $x \in \{0, 1\}^{|X|}$  such that  $x$  is consistent with  $\rho$  and  $Ax|_I = b|_I$ . But we may satisfy the equation corresponding to the vertex  $y$  (probably we should change the value of the variable  $z$  in the assignment  $x$ ), we get a contradiction. Thus  $\delta(I) \subseteq \text{vars}(\rho)$ , then  $\delta(I \cup \text{Cl}(\text{vars}(\rho))) \subseteq \text{vars}(\rho)$  and  $|I \cup \text{Cl}(\text{vars}(\rho))| \leq r$  since item 1 of Proposition 4.1  $|\text{Cl}(\text{vars}(\rho))| \leq |\rho| < r/2$ . The maximality of  $\text{Cl}(\text{vars}(\rho))$  implies  $I \subseteq \text{Cl}(\text{vars}(\rho))$  and the Lemma follows.  $\square$

## 5 Refined splitting tree

We prove the following Lemma in the Section 6.

**Lemma 5.1.** There exists an algorithm that given  $n \in \mathbb{N}$  as an input in polynomial in  $n$  time returns an  $(r, d, c)$ -boundary expander  $G$  with  $n$  vertices in each part with nonsingular adjacency matrix, where  $r = \Omega(n)$ ,  $c > 2$ ,  $d$  is a constant and degrees of all vertices from  $X$  are bounded by a constant  $D$ .

Let  $G$  be a graph from Lemma 5.1 and  $A$  be its adjacency matrix. Let  $x_1 := \alpha$  be such a substitution to the variable  $x_1$  that makes the system  $Ax = b$  unsatisfiable. Let formula  $\Phi$  encode the system of equalities  $Ax = b$  after the substitution  $x_1 := \alpha$ .

We consider the splitting tree  $T_\Phi$  made by algorithm  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{1}}$  on the input  $\Phi$ . We may assume that every vertex of  $T_\Phi$  has a partial substitution that consists of all substitutions made on the path from the root to the current vertex. We also include  $x_1 := \alpha$  in all such substitutions.

We go through all vertices of the tree  $T_\Phi$  in the order of increasing of their depth. For some vertices we will delete the subtree rooted by them from  $T_\Phi$ . If a vertex is deleted we will not consider it latter. Suppose we consider a vertex  $v$ ; if the partial substitution  $\rho_v$  that corresponds to  $v$  is not locally consistent, then we remove the subtree rooted with  $v$  from  $T_\Phi$ . We denote the resulting tree by  $T'_\Phi$ .

**Lemma 5.2.** 1) The length of every path in  $T'_\Phi$  from the root to a leaf is at least  $r/2 - 2$ .  
2) At least one half of first  $r/2 - 3$  vertices on every path from the root to a leaf in  $T'_\Phi$  are splitting points (that is they have two direct descendants).

*Proof.* 1) Since  $c > 2$  we get  $\text{Cl}(\{x_1\}) = \emptyset$  and therefore the substitution  $\{x_1 := \alpha\}$  is locally consistent. Let  $\rho$  be a locally consistent substitution that corresponds to a vertex  $v$  in the tree  $T_\Phi$ . Let  $x_v$  be a splitting variable in  $v$  and  $|\rho| < r/2 - 1$ ; item 1 of Proposition 4.1 implies  $\text{Cl}(\text{vars}(\rho) \cup \{x_v\}) < r/2$ . Since  $\rho$  is locally consistent, there exists such  $\alpha_v$  that  $\rho \cup \{x_v := \alpha_v\}$  is locally consistent. Finally we note that a leaf of the tree  $T_\Phi$  can't correspond to a locally consistent substitution since  $\Phi$  is unsatisfiable.

2) Let  $v$  be some vertex of  $T'_\Phi$  that belongs to the path from the root of length at most  $r/2 - 3$  and  $u$  be the vertex that belongs to the path from the root to a leaf that contains  $v$ , and the length of the path from the root to  $u$  equals  $\lceil r/2 \rceil - 1$ . Let's  $|\rho_v| \leq r/2 - 2$  and  $\rho_v$  is locally consistent. Let's  $\rho'_v = \rho_v \cup \{x_v := \alpha_v\}$  is not locally consistent. The latter means that the value of  $x_v$  follows from the substitution  $\rho_v$  and equalities that correspond to  $\text{Cl}(\text{vars}(\rho'_v))$ . The item 3 of Proposition 4.1 implies that  $\text{Cl}(\text{vars}(\rho'_v)) \subseteq \text{Cl}(\text{vars}(\rho_u))$ . We split the set  $\text{vars}(\rho_u)$  into  $P \cup Q$ , where  $P$  corresponds to variables in splitting vertices (i.e. the corresponding vertex has two direct descendents) and  $Q$  corresponds

to variables with unique descendant. The values of variables from  $Q$  are followed from the value of variables  $P$  and equalities corresponding to the set  $Cl(vars(\rho_u))$ . Note that equalities corresponding to the partial substitution  $\rho_u$  are linearly independent. Therefore  $|Q| \leq |Cl(vars(\rho_u))| \leq \frac{|vars(\rho_u)|}{c} < \frac{|vars(\rho_u)|}{2}$ .  $\square$

**Corollary 5.1.** The size of  $T'_\Phi$  (and therefore  $T_\Phi$ ) is at least  $2^{r/4-2}$ .

Let  $K$  be the upper bound on the number of bits of right hand side of linear system that procedures **A** and **C** can open per step. Now we construct the system of myopic copies for the formula  $\Phi$ . Let's consider the tree  $T'_\Phi$ ; on every path from the root to a leaf we select a vertex such that among its ancestor there are exactly  $N$  splitting points, where  $N \leq \frac{r/4-2}{K}$ . Lemma 5.2 implies that the distance from every selected vertex to the root is at most  $2N$  and the number of selected vertices equals  $2^N$ . Let's denote the set of selected vertices by  $\{v_1, v_2, \dots, v_{2^N}\}$ . We consider the execution of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  along the path from the root to  $v_i$ . Along this path procedures **A** and **C** open at most  $2NK$  bits of  $b$ . Since  $2NK < r/2$  and the substitution  $\rho_{v_i}$  is locally consistent there exists a vector  $b'$  such that the system  $Ax = b'$  has a solution that is consistent with  $\rho_{v_i}$  and  $b$  and  $b'$  agree on all bits open along the path. Let  $\Phi_i$  be a formula that encodes the linear system  $Ax = b'$  after the substitution  $x_1 := \alpha$ .

We still have to prove that the uniform distribution on formulas  $\Phi_i$  is polynomial time samplable. Assume that there exists a polynomial time algorithm that given a vertex from  $T_\Phi$  decides whether it has one or two direct descendants in the tree  $T'_\Phi$  and if it has only one descendant it says which one. Under this assumption we may generate uniform distribution on the set  $\{v_1, v_2, \dots, v_{2^N}\}$ . Namely we simulate the running of algorithm  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{1}}$  in the following way: the procedure **A** chooses a variable for splitting. If the current vertex is a splitting point, then we choose a value for the variable uniformly at random, otherwise we choose the value that leads to the descendant in  $T'_\Phi$ . We stop after  $N$  splitting points; let  $v_i$  be the vertex where we stopped. We also should save the bits of  $b$  opened by procedures **A** and **C**. Based on this information it is not hard to construct the formula  $\Phi_i$  by Gaussing elimination.

Now we have to describe a way to determine whether a given vertex is a splitting point or not and if it is not a splitting point to find a correct descendant. Unfortunately it is not so obvious since it is not clear whether we may calculate a closure in polynomial time. Other idea is to check locally consistentness straightforward and to find the minimal unsatisfiable part of linear system; unfortunately in general the searching for the minimal unsatisfiable part of linear system is NP-hard problem (it is also known as minimal codeword problem) and even the best known approximate solution [APY09] is not enough for our purposes.

However Lemma 4.1 implies that the substitution is locally consistent if it is consistent with every set of rows of size at most  $r/2$ . If the substitution is not locally correct then it contradicts the set of rows corresponding to the closure and therefore to any superset of the closure. So it is sufficient to construct in polynomial time a superset of the closure of size at most  $r/2$ . To do this we show that the closure of the set  $J$  is contained in a relatively small neighborhood of the set  $J$ . And this neighborhood is an easy computable superset of  $Cl(J)$ .

Let  $\Gamma^k(J)$  denote the set of all vertices connected with  $J$  by a path of a length at most  $k$ .

**Lemma 5.3.** If  $|J| < r/2$ , then  $Cl(J) \subseteq \Gamma^k(J)$ , where  $k = O(\log |J|)$ .

*Proof.* Let's  $I = Cl(J)$ . We split  $I$  into disjoint parts  $I_1 = I \cap \Gamma^1(J)$ ,  $I_{j+1} = \left(I \setminus \bigcup_{i=1}^j I_i\right) \cap \Gamma^{2^{j+1}}(J)$ ; let  $I_\ell$  be the last nonempty set. We denote  $S_i = \bigcup_{j=i}^\ell I_j$ . Proposition 4.1 implies that  $|S_i| \leq |I| < r/2$ . Since  $\delta(I) \subseteq J$ , then for  $i \geq 2$  the following is satisfied  $\delta(S_i) \subseteq \Gamma(I_{i-1})$ . Hence  $c|S_i| \leq d|I_{i-1}|$ . The latter inequality implies  $|S_i| \frac{c}{d} \leq |I_{i-1}|$ , hence  $|S_i|(1 + \frac{c}{d}) \leq |S_{i-1}|$ . And therefore  $1 \leq |I_\ell| = |S_\ell| \leq \frac{|J|}{(1 + \frac{c}{d})^{\ell-1}}$ , hence  $\ell \leq \log_{1+c/d} |J| + 1$ .  $\square$

By Lemma 5.1 degrees of all vertices from  $G$  are bounded by a constant, therefore we may conclude the following corollary.

**Corollary 5.2.** There exists  $\delta > 0$  such that if  $|J| < n^\delta$ , then the inequality  $|\Gamma^k(J)| < r/2$  is satisfied.

Alltogether we prove the following statement.

**Lemma 5.4.** There exists a polynomial time algorithm that given  $n \in \mathbb{N}$  returns an unsatisfiable formula  $\Phi$ ; there exist a constant  $\delta$  such that for all myopic polynomial time procedures  $\mathbf{A}$  and  $\mathbf{C}$  there exists a system of myopic copies for  $\Phi$  of size  $2^N$  with myopic representatives  $\Phi_1, \Phi_2, \dots, \Phi_{2^N}$  in the splitting tree of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{1}}$  for any heuristic  $\mathbf{B}$ , where  $N = \min\{n^\delta, \frac{r/4-2}{K}\}$  and  $K$  is a total number of clauses that procedures  $\mathbf{A}, \mathbf{C}$  may request per step. Moreover the uniform distribution on the set  $\{\Phi_1, \Phi_2, \dots, \Phi_{2^N}\}$  is samplable in polynomial in  $n$  time.

Lemma 3.1 and Lemma 5.4 imply

**Theorem 5.1.** There exists a polynomial time algorithm that outputs unsatisfiable formula  $\Phi^{(n)}$  in polynomial in  $n$  time. There exists  $\delta > 0$  such that for every myopic polynomial time procedures  $\mathbf{A}$  and  $\mathbf{C}$  there exists polynomial time samplable ensemble of distributions  $R_n$  concentrated on satisfiable formulas such that if for some procedure  $\mathbf{B}$  and some  $\epsilon > 0$  the inequality  $\Pr_{\phi \leftarrow R_n}[\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$  is satisfied, then running time of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}(\Phi)$  is at least  $(1 - \epsilon)2^N$ , where  $N = \min\{n^\delta, r/K\}$  and  $r = \Omega(n)$ .

## 5.1 Universal distribution

Based on the fact that formula  $\Phi^{(n)}$  doesn't depend on procedures  $\mathbf{A}$  and  $\mathbf{C}$  and that sampling time of  $R_n$  is bounded by a polynomial in running time of  $\mathbf{A}$  and  $\mathbf{C}$ , we construct the universal distribution that would dominate all others distributions. Let  $(\mathcal{R}^{(i)}, c_i)$  be an enumeration of all samplers  $\mathcal{R}^{(i)}$  with time bounded by  $n^{c_i}$  for all distributions given by Theorem 5.3 for all  $DPLL$  algorithms based on myopic polynomial time procedures  $\mathbf{A}$  and  $\mathbf{C}$ . We construct a sampler  $\mathcal{Q}$ , that defines the ensemble of distributions  $Q_n$ .  $\mathcal{Q}$  on the input  $1^n$  with probability  $\frac{1}{2^i}$  outputs the answer of the sampler  $\mathcal{R}^{(i)}(1^{\lfloor n^{1/c_i} \rfloor})$  executed on at most  $n^{c_i}$  steps for all  $1 \leq i \leq n$ , and with probability  $\frac{1}{2^n}$  outputs some fixed satisfiable formula  $\phi_0$ .

**Theorem 5.2.** For every polynomial time myopic procedures  $\mathbf{A}$  and  $\mathbf{C}$  there exists such positive  $\epsilon$  that for every procedure  $\mathbf{B}$  algorithm  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  for all  $n$  errs on a random formula distributed according to  $Q_n$  with probability less than  $\epsilon$ , then for all large enough  $n$  the running time of  $\mathcal{D}_{\mathbf{A}, \mathbf{B}, \mathbf{C}}$  on the input  $\Phi^{(n)}$  is at least  $2^{N-1}$ , where  $N = \min\{n^\delta, r/K\}$  and  $r = \Omega(n)$ ,  $\delta$  is a positive constant.

*Proof.* Let  $\mathcal{R}_i$  be a sampler that corresponds to procedures **A** and **C** and its running time is bounded by  $n^{c_i}$ . Then for all  $n \geq i$  the sampler  $\mathcal{R}^{(i)}$  is present in the enumeration in the definition of  $\mathcal{Q}$ . Let  $\epsilon = \frac{1}{2^{i+1}}$ ; for  $i \geq n$  the algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  works correctly on a random input according to  $R_{\lfloor n^{1/c_i} \rfloor}^{(i)}$  with probability at least  $\frac{1}{2}$ , where  $R_n^{(i)}$  is an ensemble of distributions generated by  $\mathcal{R}^{(i)}$ . Let  $m = \lfloor n^{1/c_i} \rfloor$ ; then the algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  runs on the input  $\Phi_m$  at least  $2^{N-1}$  steps, where  $N = \min\{m^\delta, r/K(m)\}$  and  $r = \Omega(m)$ ,  $\delta$  is a positive constant from Theorem 5.3. We are done since  $m$  goes through all large enough natural numbers.  $\square$

## 5.2 Lower bound on satisfiable instances

If we restrict procedure **B** to be myopic then it is possible to prove the result that is similar to the Theorem 5.3 but for satisfiable formulas  $\Phi^{(n)}$ .

**Theorem 5.3.** There exists  $\delta > 0$  such that for every myopic polynomial time procedures **A**, **B** and **C** there exists a polynomial time algorithm that outputs satisfiable formula  $\Psi^{(n)}$  in polynomial in  $n$  time and there exists polynomial time samplable ensemble of distributions  $R_n$  concentrated on satisfiable formulas such that if for some  $\epsilon > 0$  the inequality  $\Pr_{\phi \leftarrow R_n}[\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\phi) = 1] \geq 1 - \epsilon$  is satisfied, then running time of  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}(\Psi)$  is at least  $(1 - \epsilon)2^N$ , where  $N = \min\{n^\delta, r/K\}$  and  $r = \Omega(n)$ .

*Proof.* Consider the matrix  $A$  that was used in the construction of formula  $\Phi^{(n)}$  in the proof of the Theorem 5.3. Let  $b_0 = 0^n$ ; we execute algorithm  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  on the formula  $Ax = b_0$ . Consider the first substitution that  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  makes. W.l.o.g. we assume that the first substitution is  $x_1 := \alpha_1$  and the algorithm opens bits from a set  $Z_1$ . We choose formula  $\Psi^{(n)}$  as satisfiable formulas of the form  $Ax = b$  where all bits of  $b$  from  $Z_1$  are zeros and formula  $Ax = b$  after the substitution  $x_1 := \alpha_1$  becomes unsatisfiable. The substitution  $x_1 := \alpha_1$  is obviously locally consistent since  $c > 2$ ; therefore such satisfiable formula  $\Psi^{(n)}$  indeed exists.

Now we may take the distribution  $R_n$  from the Theorem 5.3 for formula  $\Phi^{(n)}$  that correspondes to  $Ax = b$  after the substitution  $x_1 := \alpha_1$ . Finally we note that running time of  $\mathcal{D}_{\mathbf{A},\mathbf{B},\mathbf{C}}$  on the input  $\Psi^{(n)}$  is at least its running time on the input  $\Phi^{(n)}$  since the first substitution that the algorithm makes is  $x_1 := \alpha_1$ .  $\square$

## 6 Construction of expander

In this section we give an explicit construction of  $(r, d, c)$ -boundary expanders with  $n$  vertices in every part, where  $r = \Omega(n)$ ,  $c > 2$  that have two additional properties:

- degrees of all vertices from  $X$  are bounded by a constant;
- the adjacency matrix of this graph is nonsingular.

**Definition 6.1.** The bipartite graph  $G$  with parts  $X$  and  $Y$  is an  $(r, d, c)$ -expander, if degrees of all vertices from  $Y$  do not exceed  $d$  and for every set  $I \subseteq Y$ ,  $|I| \leq r$  the following holds  $|\Gamma(I)| \geq c|I|$ , where  $\Gamma(I)$  denotes the set of all vertices that are adjacent with at least one vertex from  $I$ .

**Lemma 6.1** ([AHI05]). Every  $(r, d, c)$ -expander is a  $(r, d, 2c - d)$ -boundary expander.

We say that a family of graphs  $G_n$  is explicit if it is possible to construct  $G_n$  in polynomial in  $n$  time.

**Theorem 6.1.** [CRVW02] For every  $\epsilon > 0$  there exists  $k \geq 1$  and exists an explicit construction of an  $(r, d, (1 - \epsilon)d)$ -expander with sizes of parts  $|X| = m$  and  $|Y| = n$ , where  $r = \frac{n}{kd}$ ,  $d = \text{polylog}(\frac{n}{m})$ .

**Corollary 6.1.** For every large enough  $d$  and every  $n$  there exists an explicit construction of  $(\frac{n}{kd}, d, 0.95d)$ -expanders with  $|X| = n$ ,  $|Y| = 2n$ , where  $k$  is a constant.

**Lemma 6.2.** For every large enough  $d$  and every  $n$  there exists an explicit construction of  $(r, d, 0.75d)$ -expanders with  $|X| = |Y| = n$ ,  $r = \Omega(n)$  and degrees of all vertices from  $X$  are at most  $20kd$ .

*Proof.* We construct graph  $G$  according to Corollary 6.1. We find the set of vertices  $T \subseteq X$  with degrees greater than  $20kd$ . Since the number of edges is at most  $2nd$  we have  $|T| \leq \frac{2nd}{20kd} = \frac{n}{10k}$ . If  $|T| < \lfloor \frac{n}{10k} \rfloor$ , we add several vertices in  $T$  to make such that  $|T| = \lfloor \frac{n}{10k} \rfloor$ . Let  $Z$  be a set of vertices from  $Y$  those are connected with  $T$  by at least  $\frac{d}{5}$  edges. Let  $|Z| \geq \frac{n}{kd}$  and  $Z' \subseteq Z$ ,  $|Z'| = \lfloor \frac{n}{kd} \rfloor$ . There are at least  $\frac{1}{5}|Z'|d$  edges connected  $Z'$  and  $T$ . Therefore  $|\Gamma(Z')| \leq |T| + \frac{4}{5}d|Z'| \leq 0.9\frac{n}{k} < 0.95d|Z'|$  for  $n$  large enough; this contradicts with the expansion property. Hence for  $n$  large enough we have  $|Z| < \frac{n}{kd}$ . We remove from the graph sets  $T$  and  $Z$  and several vertices from  $Y$  to make sizes of  $X$  and  $Y$  equal to  $\lceil n(1 - 1/kd) \rceil$ . Since we remove several edges that are adjacent with some vertices from  $Y$ , the resulting graph is an  $(\frac{n}{kd}, d, 0.75d)$ -expander.  $\square$

**Lemma 6.3.** For every  $d$  large enough and every  $n$  there exists an explicit construction of  $(r, d + 2, 0.75d)$ -expanders with  $|X| = |Y| = n$ ,  $r = \Omega(n)$ , degrees of all vertices from  $X$  are at most  $20kd + 2$  and the adjacency matrix of  $G$  is nonsingular over  $\mathbb{F}_2$ .

*Proof.* First of all we prove an auxiliary statement

**Proposition 6.1.** Let  $a = (\alpha_1, \alpha_2, \dots, \alpha_n) \in \{0, 1\}^n$  and  $\sum_{i=1}^n \alpha_i = 0 \pmod{2}$ . Then vectors  $b_1 = a + (1, 0, \dots, 0)$ ,  $b_2 = a + (0, 1, 0, \dots, 0)$ ,  $\dots$ ,  $b_n = a + (0, \dots, 0, 1)$  are linearly independent over  $\mathbb{F}_2$ .

*Proof.* The sum of the elements of every  $b_i$  is odd, hence the sum of the odd number of  $b_i$  can't be zero. Let  $e_i$  be  $i$ -th vector of a standard basis (the unique one is situated on the  $i$ -th place).  $b_{i_1} + \dots + b_{i_{2\ell}} = 2\ell \cdot a + e_{i_1} + \dots + e_{i_{2\ell}} = e_{i_1} + \dots + e_{i_{2\ell}} \neq 0$ , since the sum of the basis vectors can't be zero.  $\square$

We construct the graph  $G$  according to Lemma 6.3. We consider some matching between vertices of  $X$  and  $Y$  (it is not necessary that matched variables are connected by edge). For every vertex from  $Y$  with odd degree we add edge to its pair. Finally we increase degrees in  $X$  and in  $Y$  by at most 1 and all degrees in  $Y$  become even.

We consider the adjacency matrix of new bipartite graph. Let  $k$  be its rank and  $k < n$ . We consider  $k$  linear independent rows of this matrix; we will not modify them. We apply the following operation to every of remaining  $n - k$  rows: we add 1 to one of the positions in the row in order to increase the rank exactly by one (this is possible by

Proposition 6.1). At the end we add to the initial matrix a new matrix with  $n - k$  ones such that the rank of the sum is  $n$ . The latter means that all added ones were situated in different columns.  $\square$

**Corollary 6.2.** The graph from the statement of the Lemma by Lemma 6.1 is a  $(r, d + 2, \frac{1}{2}d - 2)$ -boundary expander.

**Acknowledgments** The authors thank Edward A. Hirsch for the statement of the problem and also thank Ilya Posov, Elena Ikonnikova and anonymous reviewers for useful comments.

## References

- [ABBO<sup>+</sup>09] Michael Alekhnovich, Allan Borodin, Joshua Buresh-Oppenheim, Russell Impagliazzo, Avner Magen, and Toniann Pitassi. Toward a model for backtracking and dynamic programming. *Computational Complexity*, pages 1–62, 2009. 10.1007/s00037-011-0028-y.
- [AHI05] Michael Alekhnovich, Edward A. Hirsch, and Dmitry Itsykson. Exponential lower bounds for the running time of DPLL algorithms on satisfiable formulas. *J. Autom. Reason.*, 35(1-3):51–72, 2005.
- [APY09] Noga Alon, Rina Panigrahy, and Sergey Yekhanin. Deterministic approximation algorithms for the nearest codeword problem. In *Proceedings of the 12th International Workshop and 13th International Workshop on Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques, APPROX '09 / RANDOM '09*, pages 339–351, Berlin, Heidelberg, 2009. Springer-Verlag.
- [CEMT09] James Cook, Omid Etesami, Rachel Miller, and Luca Trevisan. Goldreich’s one-way function candidate and myopic backtracking algorithms. In *Proceedings of TCC*, pages 521–538. Springer-Verlag, 2009.
- [CRVW02] M. Capalbo, O. Reingold, S. Vadhan, and A. Wigderson. Randomness conductors and constant-degree expansion beyond the degree/2 barrier. In *Proceedings of the 34th Annual ACM Symposium on Theory of Computing*, pages 659–668, 2002.
- [Gol00] Oded Goldreich. Candidate one-way functions based on expander graphs. Technical Report 00-090, Electronic Colloquium on Computational Complexity, 2000.
- [GSTS07] Dan Gutfreund, Ronen Shaltiel, and Amnon Ta-Shma. If NP languages are hard on the worst-case, then it is easy to find their hard instances. *Computational Complexity*, 16(4):412–441, 2007.
- [HIMS11] Edward A. Hirsch, Dmitry Itsykson, Ivan Monakhov, and Alexander Smal. On optimal heuristic randomized semidecision procedures, with applications

to proof complexity and cryptography. *Theory of Computing Systems*, 2011. Extended abstract appeared in the proceedings of STACS-2010.

- [IS11] D. Itsykson and D. Sokolov. The complexity of inversion of explicit Goldreichs function by DPLL algorithms. In *Proceedings of CSR 2011*, volume 6651 of *Lecture Notes in Computer Science*, pages 134–147. Springer, 2011.
- [Its10] D. Itsykson. Lower bound on average-case complexity of inversion of Goldreich function by drunken backtracking algorithms. In *Proceedings of III International Computer Science Symposium in Russia*, volume 6072 of *Lecture Notes in Computer Science*, pages 204–215. Springer, 2010.
- [Mil09] Rachel Miller. Goldreich’s one-way function candidate and drunken backtracking algorithms. Master’s thesis, University of Virginia, 2009. Distinguished Majors Thesis.
- [Tse68] G. S. Tseitin. On the complexity of derivation in the propositional calculus. *Zapiski nauchnykh seminarov LOMI*, 8:234–259, 1968. English translation of this volume: Consultants Bureau, N.Y., 1970, pp. 115–125.
- [Urq87] A. Urquhart. Hard examples for resolution. *JACM*, 34(1):209–219, 1987.