Electronic Colloquium on Computational Complexity, Report No. 3 (2002)

A Gap in Average Proof Complexity

Eli Ben-Sasson [*] Yonatan Bilu [†]

December 24, 2001

Abstract

We present the first example of a natural distribution on instances of an NP-complete problem, with the following properties. With high probability a random formula from this distribution (a) is unsatisfiable, (b) has a short proof that can be found easily, and (c) does not have a short (general) resolution proof. This happens already for a very low clause/variable density ratio of $\Delta = \log n$ (n is the number of variables). This is the first example of such a natural distribution for which general resolution proofs are not the best way for proving unsatisfiability of random instances. Our result gives hope that efficient proof methods might be found for random 3-CNFs with small clause density (significantly less than $\sqrt{n}$).

Keywords: Proof Complexity, resolution.

[*] Department of Electrical Engineering and Computer Science, Harvard University, Cambridge, MA. eli@eecs.harvard.edu

[†] Institute for Electrical Engineering and Computer Science, Hebrew University, Givat-Ram, Jerusalem, Israel. johnblue@cs.huji.ac.il

ISSN 1433-8092

1 Introduction

Assuming $\mathbf{P} \neq \mathbf{NP}$ any algorithm solving an $\mathbf{NP}$ complete problem must operate miserably on some inputs. This assumption still does not tell us whether a *random* instance is easy or hard. The investigation of properties of various natural distributions on instances of NP-complete problems has received much attention in recent years, with most attention focusing on the following model. Select uniformly at random $\Delta \cdot n$ clauses, each of size k (usually $k = 3$), from the set of all $\binom{n}{k} \cdot 2^k$ possible clauses. We denote this distribution by $\mathbb{C}_\Delta^{k,n}$, where Δ is called the *clause density*, and denote by $\mathcal{C} \sim \mathbb{C}_\Delta^{k,n}$ a random CNF from this distribution. Many interesting facts are known about $\mathbb{C}_\Delta^{k,n}$, and in our brief survey we will only discuss $\mathbb{C}_\Delta^{3,n}$, noting that all results can be generalized to larger k. It is known that as Δ goes from $\leq 3.145\ldots$ [1] to $\geq 4.5793\ldots$ [15] the fraction of satisfiable formulas in $\mathbb{C}_\Delta^{3,n}$ goes from $1 - o(1)$ to $o(1)$, and that the threshold is sharp [12]. Knowing this, one may ask what is the range of Δ for which the satisfiability problem is hard to solve. When $\Delta < 3.145$ this amount to finding a satisfying assignment to a formulas, whereas if $\Delta > 4.5793$ this is equivalent to finding a short proof of unsatisfiability.

In recent years, many empirical heuristics have been developed for addressing the satisfiability problem. These methods have also grave implications on practical areas such as constraint satisfaction [18] and symbolic model checking [8] Some of these methods try to randomly find a satisfying assignment, by starting from some assignment and trying to improve it by local increments (e.g. at any step flip a bit as to minimize the number of unsatisfied clauses, see [18]). Naturally, when the clause density is very large, these methods fail with high probability, and moreover, one cannot be convinced that an assignment does not exist. In the unsatisfiable range, one usually reverts to a proof system, and seeks a short proof that the formula is unsatisfiable. A very natural framework for this problem is that of *resolution*. Many currently used automated theorem provers use DLL procedures [11], that rely on a weak form of resolution, called *treelike resolution*. Thanks to some recent developments, we can now say that for some clause densities, there are proof methods that exponentially outperform treelike resolution. This inefficiency result is the combination of tight lower bounds for the size of treelike resolution, that behave like $\exp(\frac{n}{\Delta})$ [6], and the recent elegant result of Friedman and Geordt [13], showing that using a different proof method, one can easily find proofs of unsatisfiability for $\Delta > \sqrt{n}$.

Our knowledge of resolutions efficiency with respect to random instances is less clear. The best current lower bounds for resolution proof size of a random $\mathcal{C} \sim \mathbb{C}_\Delta^{3,n}$ are roughly $\exp(\frac{n}{\Delta^2})$ [4, 7], and we currently do not know of any other proof method that outperforms resolution in the range $\Delta < \sqrt{n}$. This might lead one to suggest that random instances within the range $4.5793 < \Delta < \sqrt{n}$ might be hard for *all* proof systems. In this paper we provide initial evidence contradicting this belief.

Although the distribution $\mathbb{C}_\Delta^{3,n}$ has received by far the most interest, looking at similar distributions is often fruitful, and interesting in its own right. Insights about such closely related distributions often lead to a better understanding of $\mathbb{C}_\Delta^{3,n}$ itself. For instance, the recent upper bound for random 3-CNFs with clause density $\Delta > \sqrt{n}$ [13] was motivated and preceded by upper bounds for random 4-CNFs with clause density $\Delta > n^2$ [14]. Another example is the recent work of [3] examining a mixed distribution of 2 and 3 clauses, which yields interesting lower bounds on the run time of various SAT solvers. In this paper, we look at a natural distribution over instances of 4-EH SAT (see definition in section 2.1) [1]. For this distribution over instances of an NP-complete problem, very similar to $\mathbb{C}_\Delta^{k,n}$, we prove the following results:

1. For $\Delta > 0.71$, a random instance over n variables and Δn constraints in unsatisfiable with high

[1] Similar distributions over instances of the closely related problems of NAESAT and 1-in-k-SAT were recently considered in [2], where the exact threshold constant of these problems was determined.

probability.

2. With high probability there exist short proofs for random instances when $\Delta > \log n$. Moreover, these proofs can be found in polynomial time.

3. With high probability the minimal resolution proof size of a random instance is $\exp(\frac{n}{\log^{O(1)} n})$, for $\Delta = \log n$.

This is the first example of a natural distribution on instances of an NP-complete language, for which a proof system exists that exponentially outperforms resolution. This relative inefficiency of resolution, and the complementary efficiency of the other proof system, occur for a very low clause density of $\log n$. Our work gives hope for finding similar efficient proof systems for random 3-CNFs with clause density significantly smaller than $\sqrt{n}$.

2 Exactly Half SAT has Short Proofs

In this section we define our natural distribution, and show that even for very low clause density, there exist short proofs of unsatisfiability, that can easily be found. We start by defining our distribution, and then proceed to prove the upper bound. The main result of this section is the upper bound, appearing in theorem 2.1.

2.1 Exactly Half SAT - Definitions and Basic Facts

The Exactly Half SAT problem is the following constraint satisfiability problem: given a set of clauses, each containing an even number of literals, is there an assignment that satisfies exactly one half of the literals in each clause?
The 2k-EH SAT problem is the same problem, where all clauses contain exactly 2k literals. The focus of this work is on 4-EH SAT .

We first note that by a reduction from 3-Not-All-Equal SAT (also a conclusion from [19]) one gets:

Lemma 2.1 *For any $k > 1$, 2k-EH SAT is NP-Complete.*

Proof: We start by showing that 4-EH SAT is NP-Complete. Given a 3-NAESAT formula with m clauses, add m new variables, one to each clause. If the original formula is satisfiable, then use the same satisfying assignment for the old variables, and set the value of the new variable so that exactly one half of the literals are satisfied. This is possible since each clause has either exactly two literals set to "TRUE", or exactly two literals set to "FALSE", and new variables appear only in one clause.
Conversely, consider a satisfying assignment for the constructed 4-EH SAT formula. For each clause, the original literals can not be all assigned the same value, so this assignment also satisfies the original 3-NAESAT formula.
Now we can show a reduction from 4-EH SAT to 2k-EH SAT. Given a 4-EH SAT formula with m clauses, add $k-2$ new variables to each clause. Add each new variable twice - once negated, and once unnegated. Since of each such pair of literals exactly one is set to "TRUE", the new formula is satisfiable iff the original one is. □

Now consider a random 4-EH SAT formula with n variables and $m = \Delta n$ clauses, picked according to $\mathbb{C}_{\Delta}^{4,n}$. As with CNF formulas, a linear number of clauses guarantees that the generated formula is, with high probability, unsatisfiable:

Lemma 2.2 *A random 4-EH SAT formula with n variables and $0.71 \cdot n$ clauses is, w.h.p., not satisfiable.*

Proof: A given assignment satisfies a randomly chosen clause with probability $\frac{6}{16}$, and thus the probability that is satisfies all m clauses is $(\frac{3}{8})^m$. By the union bound, the probability that some assignment satisfies all m clauses is at most $2^n(\frac{3}{8})^m$. picking $m \geq 0.71n$, makes this probability exponentially small. □

2.2 Finding Short Proofs of Unsatisfiability

Using random graph theory arguments, one can show that w.h.p. there are short proofs of unsatisfiability for random 2k-EH SAT instances, with n variables and $\Omega(n^k)$ clauses: Set $N = 2^k\binom{n}{k}$, and consider the random graph on N vertices defined by a 2k-EH SAT in the following way: The vertices of the graph are all the subsets of k literals. For each clause in the formula, pick at random k of its literals, and put an edge in the graph between the vertex representing this set, and that representing the remaining literals. If the underlying formula is picked at random, then this, in effect, is a random graph with m edges picked at random.

Lemma 2.1 *If a random graph constructed in the above manner is connected, then the underlying formula is unsatisfiable.*

Proof: For a given assignment, say that a vertex in the graph is a pure if the satisfying assignment sets the same value to all the literals identified with the vertex. Notice that a necessary condition for an assignment to satisfy a 2k-EH SAT formula, is that all the neighbors of a pure vertex are themselves pure. Thus, if the graph is connected, then for a satisfying assignment either all the vertices are pure, or none of them are. However, neither can be the case, since obviously for *any* assignment, exactly $2^{-(k-1)}$ fraction of the vertices are pure. □

From the theory of random graphs we know that for $m > N$, a random graph is, w.h.p., connected (e.g. [16]). By lemma 2.1, the connectivity of such a graph is a proof that the underlying 2k-EH SAT formula is unsatisfiable.

However, we can prove a stronger result:

Theorem 2.1 *Let f be a random 4-EH SAT formula on n variables, and $\Omega(n \log n)$ clauses, then, w.h.p., there is a polynomial-size proof showing that f is unsatisfiable. Furthermore, this proof can be found in polynomial time.*

We shall start with some definitions and a couple of lemmas. Denote the variables over which a 4-EH SAT formula is defined by $x_1, ..., x_n$, and its clauses by $C_1, ..., C_m$. We associate with a 4-EH SAT formula f a matrix $M_f \in M_{m,n}(\mathbb{R})$, and with an assignment α a vector $v_\alpha \in \mathbb{R}^n$ defined as follows:

$$(M_f)_{i,j} = \begin{cases} 1 & x_j \in C_i \text{ and } \overline{x}_j \notin C_i \\ -1 & \overline{x}_j \in C_i \text{ and } x_j \notin C_i \\ 0 & \text{otherwise} \end{cases}$$

$$(v_\alpha)_i = \begin{cases} 1 & \alpha(x_i) = \text{TRUE} \\ -1 & \alpha(x_i) = \text{FALSE} \end{cases}$$

Note that the rows of M_f contain at most four non-zero entries, and these are either 1 or -1. We shall call such vectors *4-clause vectors*.

Lemma 2.2 *For a 4-EH SAT formula f, if $rank(M_f) \geq n - O(\log n)$ then there is a short proof for whether f is satisfiable or not.*

Proof: Observe that v_α corresponds to a satisfying assignment iff $(M_f)v_\alpha = 0$. In particular, if f is satisfiable, then the kernel of M_f in not trivial. Since the rank of M_f can be computed in polynomial time, such a matrix of rank n, along with the computation of the rank, constitutes a short proof for the unsatisfiability of f.

Now assume that $K = Ker(M_f)$ is of dimension $t = O(\log n)$. Let $v_1, .., , v_t$ be a basis for K. Consider the matrix whose rows are the v_i's. It is of rank t, so we can find, in polynomial time, t independent columns. Denote by A the regular sub-matrix defined by all t rows and these t columns. Consider an assignment vector, u, and its restriction, u', to the t chosen columns. There is exactly one linear combination of the rows of A that results in u'. Clearly, this is the only linear combination of the v_i's that might sum up to u. Thus, all assignment vectors in $Ker(M_f)$ will be found by the following algorithm: Go over all 2^t $\{-1,1\}^t$ vectors. For each, find the linear combination of rows from A that produces it. Use this linear combination of the v_i's, and if it results in a $\{-1,1\}^t$ vector, then it corresponds to a satisfying assignment.

Since this can be done in polynomial time, it constitutes a short for proof for whether f is satisfiable or not. □

Lemma 2.3 *For a random 4-EH SAT formula f with $\Omega(n \log n)$ clauses, M_f is, w.h.p., of rank n.*

We will need the following technical claim:

Claim 2.4 *Let W be an l-dimensional linear subspace of $\mathbb{R}^n$, then W contains at most $\Sigma_{i=1}^4 2^i \binom{l}{i}$ 4-clause vectors.*

Proof: Let $v_1, ..., v_l$ be a basis for W. Consider the matrix whose rows are the v_i's. By Gaussian elimination we may assume, w.l.o.g., that this matrix is of the form $(I|A)$, where I is the $l \times l$ identity matrix, and A is some l by $n - l$ matrix.

Let u be 4-clause vector in W, and consider its representation as a linear combination of the basis elements: $u = \Sigma c_i v_i$. By the assumption on the first l coordinates of the v_i's, the first l coordinates of u are $c_1, ..., c_l$. Since u is a 4-clause vector, at most four of the c_i's can by non-zero, and in that case, they must be either 1 or -1. Thus, the number of 4-clause vectors in W is at most $\Sigma_{i=1}^4 2^i \binom{l}{i}$. □

Observe that what the proof relies on is that the size of the support of the vectors is constant. The same proof shows that if W is an l-dimensional linear subspace of $\mathbb{R}^n$, it contains at most $\Sigma_{i=1}^t 2^i \binom{l}{i}$ with support of size t.

Proof (of Lemma 2.3): Consider the process of picking 4-clause vectors one by one, uniformly at random. Call such a vector *novel* if it is not in the linear span of its predecessors. Let X by a random variable denoting the number of 4-clause vectors drawn, until the nth novel vector. Consider now the coupon collector's problem, with n types of coupons (see, e.g. [17] for a definition and analysis of this problem). Let Y be a random variable denoting the number of coupons drawn until all n are collected.

It is easy to see, that at a point where the coupon collector has exactly l different coupons, the chance of a coupon picked at random to be different from these l is exactly $1 - \frac{l}{n}$. Compare this to what happens when picking 4-clause vectors uniformly at random. By claim 2.4, at a point where exactly l novel vectors have been picked, the probability that the next vector is novel (i.e., that isn't within the linear span of the l novel vectors), is at least $1 - \frac{\Sigma_{i=1}^{4} 2^i \binom{l}{i}}{16\binom{n}{4}}$. A straight forward calculation shows that for a large enough n this is at least $1 - \frac{l}{n}$. Think of $X = \Sigma X_i$ and $Y = \Sigma Y_i$ as the sum of n random variables counting the number of coupons collected between consecutive novel vectors. Each of these variables has a geometric distribution, where the probability of "success" for X_i is at least as high as for the corresponding Y_i. So we have that for any number $C > 0$

$$Pr(X > C) \leq Pr(Y > C).$$

It is easily shown (e.g. in [17]) that $Pr(Y > 2n \log n) \leq n^{-1}$. Thus, if f has $2n \log n$ clauses, with probability at most n^{-1}, $rank(M_f) < n$. □

Theorem now 2.1 follows from these two lemmas.

2.3 Extensions to Exactly-(d_1, d_2)-SAT

The same arguments used in the proof of theorem 2.1 actually show something a bit stronger. Define Exactly-(d_1, d_2)-SAT to be the following constraint satisfaction problem: Given a set of clauses of size $d_1 + d_2$, is there an assignment where in each clause exactly d_1 literals are satisfied, and exactly d_2 are not? In particular, the Exactly-$(2, 2)$-SAT problem coincides with the 4-EH SAT problem.
Let d_1 and d_2 are constants such that $d_1 + d_2 > 2$, and n be large. Consider a random Exactly-(d_1, d_2)-SAT formula f on n variables and $\Omega(n \log n)$ clauses, and the corresponding matrix M_f. An assignment α satisfies f iff $(M_f)v_\alpha = (d_1 - d_2)\vec{1}$. By the observation following Lemma 2.4, and a coupon collector argument similar to Lemma 2.3, we have that w.h.p. $rank(M_f) = n$. Now define M'_f to be M_f, with the additional column $(d_1 - d_2)\vec{1}$. Clearly for any satisfying assignment α, $(M'_f)(v_\alpha| - 1) = 0$. But $\dim Ker(M'_f) \leq 1$, so as in Lemma 2.2, we can perform an extensive search for the satisfying assignment (In fact, either $Ker(M'_f) = \{0\}$, and there's nothing to check, or it is spanned by a single vector, and then we just have to check if it can be normalized to as assignment-like vector). Thus, we have that if f is a random Exactly-(d_1, d_2)-SAT instance, on n variables and $\Omega(n \log n)$ variables, then w.h.p. there is a polynomial-size proof for f being unsatisfiable, and this can proof can be found in polynomial time.

3 Exactly Half SAT Does not have Short Resolution Proofs

One of the most extensively used proof systems is *resolution*. In this section we show how the standard size lower bound techniques of [7] give exponential lower bounds on the size of resolution proofs of random instances of the 4-EH SAT formula problem with $n \log n$ clauses. We will need some preliminary results about resolution lower bounds, all of them taken from [7]. The main result of this section is a lower bound on the size of a resolution proof of an 4-EH SAT instance, when $\Delta = \log n$ (theorem 3.11).

3.1 Resolution - Definition

The *resolution* proof system is complete for the **co-NP** complete language of *unsatisfiable CNF formulas*, has essentially one derivation rule, and all lines in a proof are clauses. Here is a formal definition of this system.

A *literal* over x is either x, denoted also x^1, or $\bar{x}$, denoted also x^0. A *clause* is a disjunction of literals. A variable x *appears* in C, denoted $x \in C$, if a literal over x appears in C. A CNF formula is a set of clauses. The *resolution rule* is the following derivation rule:

Resolution Rule: Derive $E \vee F$ from $\{E \vee x, F \vee \overline{x}\}$, where E, F are any clauses, and x is any variable.

The *resolution proof system* is the sequential proof system based on the resolution rule. Let $\mathcal{C} = \{C_1, C_2 \ldots C_m\}$ be a CNF formula over n variables. A *resolution derivation* of a clause A from $\mathcal{C}$ is a sequence of clauses $\pi = \{D_1, D_2 \ldots D_S\}$ such that the last clause is A and each formula D_i is either some initial clause $C_j \in \mathcal{C}$, or is derived from previous clauses using the resolution rule. A *resolution proof,* sometimes also called a *refutation,* is a resolution derivation of the empty clause, $\mathbf{0}$. The minimal size of a proof of $\mathcal{C}$, denoted $\mathbf{S_R}(\mathcal{C})$, is the minimal number of clauses in a proof of $\mathcal{C}$. Similarly, $\mathbf{S_T}(\mathcal{C})$ is the minimal size of a treelike resolution proof. The following fundamental theorem of resolution implies that $\mathbf{S_T}(\mathcal{C}) = \mathbf{S_R}(\mathcal{C}) = \infty$ iff $\mathcal{C} \in \mathbf{SAT}$.

Theorem 3.1 (Completeness of Resolution) *Resolution is a complete proof system for CNF formulas. In other words, for any CNF $\mathcal{C}$, $\mathcal{C} \in \mathbf{co-SAT}$ iff there exists a resolution proof of $\mathcal{C}$.*

3.2 Encoding 4-EH SAT as a CNF

A *constraint* is a Boolean function. Let $\mathcal{F} = \{f_1 \ldots f_m\}$ be a set of constraints, where each constraint is over the variables $x_1, \ldots, x_n$. We say $\mathcal{F}$ is *satisfiable* iff there exists an assignment $\alpha \in \{0,1\}^n$ such that $f_i(\alpha) = 1$ for $i = 1 \ldots m$, and otherwise we call it *unsatisfiable.*

For $f(x_1, \ldots, x_n) : \{0,1\}^n \to \{0,1\}$ a constraint, let *Supp*(f) be the set of variables on which f depends, i.e. it is the minimal subset $X \subseteq \{x_1, \ldots, x_n\}$ such that any assignment to $\{0,1\}^X$ fixes f.

Definition 3.2 (Explicit Encoding of Boolean Constraints) *For $f(x_1, \ldots, x_n) : \{0,1\}^n \to \{0,1\}$ a Boolean constraint, an* explicit encoding *of f, denoted $\mathcal{C}(f)$, is some CNF formula over Supp(f) that is equivalent to f, i.e.*

$$\forall \alpha \in \{0,1\}^n \quad f(\alpha) = \mathcal{C}(f)(\alpha).$$

For $\mathcal{F} = \{f_1, \ldots f_m\}$ a set of Boolean constraints over variable set X, an explicit encoding *of $\mathcal{F}$ is*

$$\mathcal{C}(\mathcal{F}) \stackrel{\text{def}}{=} \bigcup_{i=1}^{m} \{\mathcal{C}(f_i)\}$$

where $\mathcal{C}(f_i)$ is an explicit encoding of f_i, for $i = 1 \ldots m$.

3.3 Expansion, Sensitivity and Size

We use the connection of the structure of a set of constraints to the proof size of this set.

Definition 3.3 *For $\mathcal{F} = \{f_1, \ldots f_m\}$ a set of constraints, the* graph *of $\mathcal{F}$, denoted $\mathbb{G}(\mathcal{F})$, is the following bipartite graph $\mathbb{G}(\mathcal{F}) =< V \dot{\cup} U, E >$.*

1. *V is the set of constraints.*
2. *U is the set of variables.*
3. *$(f_i, x_j) \in E$ iff $x_j \in Supp(f_i)$.*

Definition 3.4 (Bipartite Expanders) *A bipartite graph $\mathbb{G} =< V \dot\cup U, E >$ is called an (r, c)-*expander *if*

$$\forall V' \subset V \quad |V'| \leq r, \;\; |N(V')| \geq (1 + c)|V'|,$$

where $N(V')$ is the set of neighbors of V'.

Definition 3.5 (Boundary Expansion) *Let $G =< V \dot\cup U, E >$ be a bipartite graph. The* boundary *of $V' \subseteq V$ is*

$$\partial V' \overset{\text{def}}{=} \{u \in U : |N(u) \cap V'| = 1\}$$

In words: the boundary *of V' is the set of* unique neighbors *of V', such that every boundary element is a neighbor of exactly one vertex of V'.*

*G is called an (r, c)-*boundary expander *if*

$$\forall V' \subseteq V, \quad |V'| \leq r \quad |\partial V'| \geq c \cdot |V'|$$

A set of constraints $\mathcal{F}$ is said to be (r, c)-boundary expanding, if $\mathbb{G}(\mathcal{F})$ is an (r, c)-boundary expander.

As one might expect, there is a connection between expansion and boundary expansion. A very good expander is also a decent boundary expander:

Claim 3.6 *If $G =< V \dot\cup U, E >$ is an (r, c)- expander of maximal degree d on the V side, then it is an $(r, 2 + 2c - d)$-boundary expander.*

Definition 3.7 (Sensitivity) *A constraint f is ℓ-*sensitive *if for any assignment α such that $f(\alpha) = 0$, and any $X \subset Supp(f)$ $|X| \leq \ell$, there is an assignment β such that $f(\beta) = 1$ and β agrees with α outside of X.*

Intuitively, a constraint is ℓ-sensitive if it can be satisfied by changing any ℓ of its variables. It is easy to see that each constraint of an 2k-EH SAT problem is k-sensitive. Thus, for the 4-EH SAT problem, each constraint is 2-sensitive. We end this section by quoting the following lower bound of [7], presenting a lower bound on proof size, as a function of the sensitivity and expansion of the input set of constraints. We use the cleaner formulation appearing in [5]

Theorem 3.8 *[7, 5] For $\mathcal{F}$ an (r, c)-boundary expanding set of ℓ-sensitive constraints, each of support $\leq d$, and $\mathcal{C}(\mathcal{F})$ an explicit encoding of it as a CNF:*

$$\mathbf{S_T}(\mathcal{C}(\mathcal{F})) = \exp(\Omega(r(c - \ell + 1) - 2d))$$

$$\mathbf{S_R}(\mathcal{C}(\mathcal{F})) = \exp\left(\Omega\left((r(c - \ell + 1) - 2d)^2 \cdot n^{-1}\right)\right)$$

When applying the above theorem to the 4-EH SAT problem, notice that each constraint is 2 sensitive, and the support size is 4. Thus we get:

Corollary 3.9 *[7] For $\mathcal{F}$ be an (r, c)-boundary expanding instance of the 4-EH SAT problem, and $\mathcal{C}(\mathcal{F})$ an explicit encoding of it as a CNF:*

$$\mathbf{S_T}(\mathcal{C}(\mathcal{F})) = \exp(\Omega(r(c-1)-8))$$

$$\mathbf{S_R}(\mathcal{C}(\mathcal{F})) = \exp\left(\Omega\left((r(c-1)-8)^2 \cdot n^{-1}\right)\right)$$

where $\mathbf{S_T}(\mathcal{C})$ is the minimal number of clauses in a treelike resolution proof, and $\mathbf{S_R}(\mathcal{C})$ is the same for regular resolution.

3.4 Lower Bound

By the previous corollary 3.9, we only need to show that with high probability a random instance of the 4-EH SAT problem with n variables, and $n \log n$ clauses, is a good boundary expander. Our main technical lemma is the following.

Lemma 3.10 *With high probability, a random 4-EH SAT instance with n variables, and $n \log n$ clauses, is an $(\Omega(\frac{n}{\log^3 n}), \frac{7}{6})$-boundary expander.*

This lemma, together with corollary 3.9, immediately gives the main result of this section:

Theorem 3.11 *With high probability, an explicit encoding of a random 4-EH SAT instance with n variables, and $n \log n$ clauses, requires resolution size $\Omega(\frac{n}{\log^6 n})$, and resolution treelike size $\Omega(\frac{n}{\log^3 n})$.*

Proof [Lemma 3.10]:

The proof of our lemma follows immediately from Lemma 3.6 and the following claim:

Claim 3.12 *A random bipartite $\mathcal{F}$ with $n \log n$ constraints and n variables is with high probability a $(n/(\log^3 n), \frac{19}{12})$-expander*

Proof: Let $\mathbb{G}(\mathcal{F})$ be the graph of $\mathcal{F}$, and set $r = n/\log^3 n$, and $c = \frac{19}{12}$. Let BAD be the event that $\mathbb{G}(\mathcal{F})$ is not an (r, c)-bipartite expander. We prove that the $\Pr[BAD]$ tends to 0 as n grows. Bound the probability of BAD by the probability that there exists a set $V' \subseteq V$, with $1 \leq |V'| \leq r$, such that $|N(V')| < (1+c)|V'|$ and then use the union bound to upper bound this probability.

Observe that there are $\binom{n \log n}{i}$ possible sets $V' \subseteq V$ of size i, and there are $\binom{n}{(1+c)i}$ possible small sets of neighbors of V'. For a given set V' of size i, and a given set U' of size $(1+c)i$, the probability that $N(V') \subseteq U'$ is

$$P_i = \left(\frac{\binom{(1+c)i}{4}}{\binom{n}{4}}\right)^i \leq \left(\frac{(1+c)i}{n}\right)^{4i}$$

Let us bound the probability of the BAD event:

$$\Pr[BAD] \;\leq\; \sum_{i=1}^{r} \binom{n \log n}{i} \cdot \binom{n}{(1+c)i} \cdot P_i$$

$$
\begin{aligned}
&\leq \sum_{i=1}^{r} (\frac{en\log n}{i})^{i} \cdot (\frac{en}{(1+c)i})^{(1+c)i} \cdot (\frac{(1+c)i}{n})^{4i} && (1)\\
&\leq \sum_{i=1}^{r} [\kappa \cdot \log n \cdot (\frac{i}{n})^{(2-c)}]^{i} && (2)\\
&\leq \sum_{i=1}^{r} [\kappa \cdot \log n \cdot (\frac{r}{n})^{(2-c)}]^{i} && \\
&\leq \sum_{i=1}^{r} [\kappa \cdot \log^{-\frac{1}{4}} n]^{i} && (3)
\end{aligned}
$$

The first inequality (1) uses the well-known estimation $\binom{a}{b} \leq (\frac{ea}{b})^{b}$, the second (2) is true for the constant $\kappa = e^{2+c} \cdot (1+c)^{3-c}$, and the last (3) follows by plugging in the values of c and r. Clearly, this geometric sum vanishes as n approaches infinity. This completes the proof of claim 3.12, and with it, lemma 3.10 is proved. □

References

[1] D. Achlioptas. Setting 2 variables at time yields a new lower bound for random 3-SAT. *In Proceedings of 32th STOC*. pp. 28-37 (2000).

[2] D. Achlioptas, A. Chtcherba, G. Istrate, C. Moore The Phase Transition in NAESAT and 1-in-k SAT In *Proceedings of the Symposium on Discrete Algorithms (SODA)* 2001, pp. 721-722.

[3] D. Achlioptas, P. Beame, M. Molloy, A Sharp Threshold in Proof Complexity, In *Proceedings of STOC 2001*, p.337-346.

[4] P. Beame, R. Karp, T. Pitassi, M. Saks. The efficiency of resolution and Davis Putnam procedures. In *Proceedings of the 30th Annual ACM Symposium on Theory of Computing (STOC-98)*, pages 561–571, New York, May 23–26 1998. ACM Press.

[5] E. Ben-Sasson Expansion in Proof Complexity. Submitted as Ph. D. Thesis to the Hebrew University, Jerusalem, 2001

[6] E. Ben-Sasson, N. Galesi. Space Complexity of Random Formulae in Resolution. In *Complexity 2001.*

[7] E. Ben-Sasson, A. Wigderson. Short Proofs are Narrow - Resolution made Simple. In *Proceedings of the* $31st$ *STOC*, pages 517–526, 1999.

[8] A. Biere, A. Cimatti, E. Clark, Y. Zhu. Symbolic model chechikng without BDDs. In Proceedings, 5th International Conference, TACAS '99 pp 193-207, Berlin, 1999. Springer-Verlag.

[9] V. Chvátal, E. Szemerédi. Many hard examples for resolutions. *Journal of the ACM* **35** pp. 759-768 (1988).

[10] O. Dubois, Y. Boufkhad, J. Mandler. Typical random 3-SAT formulae and the satisfiability problem. *In 11-th SODA* pp. 126-127 (2000).

[11] M. Davis, G. Longemann, D. Loveland. A machine program for theorem proving. In *Communications of the ACM* 7:201-215, 1960.

[12] E. Friedgut. Sharp thresholds for graph properties and the k-SAT problem, 1998. Unpublished manuscript.

[13] J. Friedman, A. Goerdt. Recognizing more unsatisfiable random 3-SAT instances efficiently. In *ICALP2001*, pp. 310-321, 2001.

[14] A. Goerdt., M. Krivelevich. Efficient recognition of random unsatisfiable k-SAT instances by spectral methods. In *Proc. STACS 2001*, LNCS.

[15] S. Janson, Y.C. Stamatiou, M. Vamvakari. Bounding the unsatisfiability threshold for 3-SAT. *Random Structure and Algorithms* (17) 2 pp.118-116 (2000).

[16] S. Janson, T. Luczak, A. Rucinski. Random Graphs, Wiley 2000.

[17] R. Motwani, P. Raghavan. Randomized Algorithms. Cambridge Univ. Press., 1995.

[18] B. Selman, H. Kautz. A new method for solving hard satisfiability problems. In *Proc. 10th National conference on Artificial Intelligence*, (AAAI-92) 440-446, 1992.

[19] T. J. Scaefer. The Complexity of Satisfiability Problems. In *Conference Record of the Tenth Annual ACM Symposium on Theory of Computing*, pages 216–226, 1978.