ПРЕПРИНТЫ ПОМИ РАН

Учредитель: Федеральное государственное бюджетное учреждение науки
Санкт-Петербургское отделение Математического института
им. В. А. Стеклова Российской академии наук

Свидетельство о регистрации средства массовой информации: ЭЛ №ФС 77-33560 от 16 октября 2008 г. Выдано Федеральной службой по надзору в сфере связи и массовых коммуникаций

Контактные данные: 191023, г. Санкт-Петербург, наб. реки Фонтанки, дом 27

телефоны:(812)312-40-58; (812) 571-57-54

e-mail: admin@pdmi.ras.ru

http://www. pdmi.ras.ru /preprint/

Заведующая информационно-издательским сектором Симонова В.Н

PDMI PREPRINT — 12/2015

Simultaneous separation of regular resolution, treelike resolution and exponent of resolution depth*

Dmitry Itsykson

St. Petersburg Department of V.A. Steklov Institute of Mathematics
of Russian Academy of Sceinces
E-mail: dmitrits@pdmi.ras.ru

December, 2015

Abstract

Every unsatisfiable CNF formula ϕ has the following parameters: the size of the minimal regular resolution proof $S_R(\phi)$, the size of the minimal treelike resolution proof $S_T(\phi)$, the minimal depth of a resolution proof $d(\phi)$. The following inequality is trivially satisfied: $2^{d(\phi)} \geq S_T(\phi) \geq S_R(\phi)$. Bonet, Esteban, Galesi and Johannsen in 2000 showed that there exists family of formulas F_n such that $S_T(F_n) = 2^{\Omega(\sqrt{S_R(F_n)})}$. Urquhart in 2011 gave a family of 3-CNF formulas H_n such that $S_T(H_n) = O(n)$ and $d(H_n) = \Omega(n/\log n)$. We observe that formulas $H_n^{\oplus}$ which is the xorification of H_n has $S_T(H_n^{\oplus}) = 2^{\Omega(n/\log n)}$ and $S_R(H_n^{\oplus}) = O(n)$; this improves the separation given by Bonet et al.

We present a family of 6-CNF formulas Φ_n such that any two values from $\{2^{d(\Phi_n)}, S_T(\Phi_n), S_R(\Phi_n)\}$ differ superpolynomially. Our formulas are based on the Pebbling contradictions on the $n \times n$ square graph. Our proof is elementary and is based on the game interpretation of the resolution depth and the xorification.

*Supported in part by Russian Foundation for Basic Research (grant 14-01-00545-a).

ПРЕПРИНТЫ
Санкт-Петербургского отделения
Математического института им. В. А. Стеклова
Российской академии наук

PREPRINTS
of the St. Petersburg Department of Steklov Institute of Mathematics

---

1 Introduction

Resolution proof system is one of the simplest and well studied proof systems for propositional logic. The interest in propositional proof systems comes from SAT solving. For today the fastest algorithms are based on CDCL (Conflict Driving Clause Learning). The running time of CDCL algorithms is close to the size of the shortest resolution proof of the input formula. [2], [12]. The classical DPLL algorithms [7, 6] that are in the basis of many others SAT solvers correspond to treelike resolution proofs.

Query complexity, decision trees and read-once branching programs. Consider some computational problem F: given an input $x \in \{0,1\}^n$ find a solution $y \in \{0,1\}^*$. We assume that there is some fixed predicate Sol such that $Sol(x,y) = 1$ iff y is a solution for x. The query complexity of a problem F (we denote it $D(F)$) is the maximum over all $x \in \{0,1\}^n$ of the minimum possible number of bits of x that is sufficient to know in order to find a correct solution for x. A decision tree for a problem F is a binary tree such that its leaves are labeled by binary strings (solutions); every internal vertex is marked by a variable from the set $\{x_1, x_2, \ldots, x_n\}$, one of outgoing edge is labeled by 1 and other by 0. For every $a \in \{0,1\}^n$ if we start a path in the root of the tree and on every step if we go from the vertex labeled by x_i along an edge that is labeled by a_i, we finally reach a leaf labeled by a correct solution for the input a. The query complexity of F may be equivalently defined as the minimal possible depth of decision trees for F. A read-once branching program for F is defined by a similar way but instead of tree we have a directed acyclic graph that has one source, outdegree 2 for all internal vertices and the sinks that are labeled by solutions. On every path from the source to a sink all variables in vertices are distinct.

Let us denote the size of the minimum decision tree for a problem F as $DT(F)$ and the size of the minimal read-once branching program for F as 1-$BP(F)$. It is easy to show that in the minimal decision tree on every path from the root to a leaf all variables are distinct, therefore we get the following inequality: 1-$BP(F) \leq DT(F) \leq 2^{D(F)}$. Consider the case where F is a problem of computing a boolean function $F : \{0,1\}^n \to \{0,1\}$. In this case it is easy to separate 1-$BP(F)$, $DT(F)$ and $D(F)$. Consider the following examples:

- $F_{n,0}(x_1, x_2, \ldots, x_n) = x_1 \vee x_2 \cdots \vee x_n$: $D(F_{n,0}) = n$, $DT(F_{n,0}) = O(n)$ and 1-$BP(F_{n,0}) = O(n)$.
- $F_{n,n}(x_1, x_2, \ldots, x_n) = x_1 \oplus x_2 \oplus \cdots \oplus x_n$, $D(F_{n,n}) = n$, $DT(F_{n,n}) = 2^n$ and 1-$BP(F_{n,n}) = O(n)$.
- For $0 \leq k \leq n$ we denote $F_{n,k}(x) = (x_1 \oplus x_2 \oplus \cdots \oplus x_k) \vee x_{k+1} \vee \cdots \vee x_n$, $D(F_{n,k}) = n$, 1-$BP(F_{n,k}) = O(n)$ and $2^k + \Omega(n) \leq DT(F_{n,k}) \leq 2^k + O(n)$.

Thus $2^{D(F_{n,\log^2 n})}$, $DT(F_{n,\log^2 n})$ and 1-$BP(F_{n,\log^2 n})$ differs superpolynomially.

Consider another computational problem, that is parametrized with unsatisfiable CNF formula ϕ with n variables. An input of the problem is an assignment to variables of ϕ; a solution is a clause of ϕ that is falsified by the assignment. For this problem the query complexity is equal to the minimal depth of a resolution proof of ϕ (we denote it $d(\phi)$), the size of the minimal decision tree for this problem is equal to the minimal size of treelike resolution proof of ϕ (we denote it $S_T(\phi)$) and the size of the minimal read-once branching program is equal to the size of the minimal regular resolution proof of ϕ (we denote it $S_R(\phi)$) [10], [9]. As in general case the following inequalities hold: $2^{d(\phi)} \geq S_T(\phi) \geq S_R(\phi)$. The superpolynomial separation for any of this two values is nontrivial. The goal of this paper is to give an example of formulas that simultaneously superpolynomially separates $2^{d(\phi)}$ and $S_T(\phi)$ and also $S_T(\phi)$ and $S_R(\phi)$.

Known results. The first exponential separation of treelike and regular resolutions was proved by Bonet, Esteban, Galesi and Johannsen [4]. The paper [3] gave a family of formulas ϕ_n such that $S(\phi_n) = O(n)$ and $S_T(\phi_n) = 2^{\Omega(n/\log n)}$, where S denotes the minimal size of general resolution proof and the [3] showed that this separation is nearly optimal. It is not clear whether ϕ_n has short regular resolution proof or not.

There is a trivial example that separates $S_T(\phi)$ and $2^{d(\phi)}$: formula $Q_n = (x_1 \vee x_2 \vee \cdots \vee x_n) \wedge (\neg x_1) \wedge (\neg x_2) \wedge \cdots \wedge (\neg x_n)$ has $d(Q_n) = n$ and $S_T(Q_n) = n + 2$. Thus the most interesting examples have bounded number of literals in clauses. Urquhart [14] gave an example of family of 3-CNF formulas H_n such that $d(H_n) = \Omega(n/\log n)$ and $S_T(H_n) = O(n)$.

There are results about separations of regular and general resolutions: superpolynomial separation [8], exponential separation [1] and near optimal separation [15].

Our results. We observe the following theorem:

Theorem 1.1. There exists a family of 6-CNF formulas H'_n with $O(n)$ variables such that $S_R(H'_n) = O(n)$ and $S_T(H'_n) = 2^{\Omega(n/\log n)}$.

Almost all key ingredients for Theorem 1.1 were already given by Urquhart [14]. Let H_n be a family of formulas from [14] such that $d(H_n) = \Omega(n/\log n)$ and $S_T(H_n) = O(n)$. Let $H_n^{\oplus}$ be a xorification of H_n; namely for every variable x in H_n we substitute it by $x' \oplus x''$, where x' and x'' are new variables and convert the resulting formula in CNF. This operation was described in the paper of Ben-Sasson, where the author refer to personal communication with Alekhnovich and Razborov. Urquhart [14] noted that $S_T(\phi^{\oplus}) \geq 2^{d(\phi)}$ for all formulas ϕ, hence we get that $S_T(H_n^{\oplus}) = 2^{\Omega(n/\log n)}$. We show that $S_R(H_n^{\oplus}) = O(n)$.

We also prove the following theorem:

Theorem 1.2. There is a family of unsatisfiable 6-CNF formulas $\Phi_{n,k}$ for all n and $1 \leq k \leq n$ such that $\Phi_{n,k}$ contains $O(n^2)$ variables and $O(n^2)$ clauses, $S_R(\Phi_{n,k}) = O(n^2), d(\Phi_{n,k}) = \Omega(n)$ and $\max\{n^2, 2^{k/2}\} \leq S_T(\Phi_{n,k}) \leq 2^{6k} + 2k(n^2 + 1)$.

We start the proof of Theorem 1.2 from the example of family of 3-CNF formulas Φ_n and prove that $d(\Phi_n) = \Theta(n)$, $S_T(\Phi_n) = \Theta(n^2)$ and $S_R(\Phi_n) = \Theta(n^2)$.

In order to construct formulas $\Phi_{n,k}$ we apply partial xorification, namely we make substitution of kind $x \leftarrow x' \oplus x''$ for some k^2 variables. By the structure of the formula we prove that $S_T(\Phi_{n,k}) = \Theta(n^2) + 2^{\Theta(k)}$, $d(\Phi_{n,k}) = \Theta(n)$ and $S_R(\Phi_{n,k}) = \Theta(n^2)$.

Comparison with other approaches. The paper [3] introduced formulas based on so-called Pebbling games. Consider a directed acyclic graph G. Let S contain all sources of G and T contain all sinks of G. In the game we may put a pebble on every vertex from S; we may put a pebble on a vertex v if there are pebbles on all immediate predecessors of v and we may remove any pebble. The goal of the game is to put a pebble to some vertex from T. A pebbling number $peb(G, S, T)$ is the minimal number of pebbles that is necessary to have simultaneously to win the game. Ben-Sasson, Impagliazzo and Wigderson [3] presented formulas based on graph G and sets S and T that has linear resolution complexity and treelike resolution complexity at least $2^{\Omega(peb(G,S,T))}$. Urquhart [14] used the same graphs but slightly different formulas; Urquhart showed that treelike resolution complexity of this formulas is $O(n)$ and resolution depth is at least $\Omega(peb(G, S, T))$. The papers [3] and [14] used graphs from the paper [11] that has $O(n)$ vertices and pebbling number $peb(G, S, T) = \Omega(n/\log n)$.

Formulas from [14] are as follows: for every vertex v of graph G we have propositional variable x_v. If $u_1, u_2, \ldots, u_k$ are immediate predecessors of v, then the formula has the clause $\neg x_v \vee x_{u_1} \vee x_{u_2} \vee \cdots \vee x_{u_k}$. Also formula contains clauses x_t for all $t \in T$ and $\neg x_s$ for all $s \in S$. Formulas that we use in the proof of Theorem 1.2 have the same structure and is based on the

specific graph. The vertices of our graph are cells of $n \times n$ square; edges connect cells with their left or lower neighbours, S consists of the topmost and rightmost cell and T consists of the leftmost lowermost cell. Our proof of lower bound on the resolution depth in Theorem 1.2 is elementary; we use game interpretation of the depth and don't use estimations of pebbling number.

The paper [4] used formulas based on pyramid graphs that are very similar to the graph of the $n \times n$ square, namely pyramid graph correspond to the half of $n \times n$ square bounded by the main diagonal. The lower bound on the pebbling number of pyramid graphs was actually proved by Cook in 1974 [5].

In Section 2 we give the definitions of the basic notions and formulate previously known results. In Section 3.1 we prove Theorem 1.1 and in Section 3.2 we prove Theorem 1.2.

2 Preliminaries

Propositional variable is one that has 0/1-value, literal is either a variable or its negation. A clause is a disjunction of literals, a CNF formula is a conjunction of clauses. A k-CNF formula is a CNF formula in which all clauses contain at most k literals. The formula is satisfiable if there exists a substitution for its variables such that the value of the formula becomes 1 (we call such substitution a satisfying assignment).

A resolution proof of formula ϕ is a sequence of clauses $C_1, C_2, \ldots, C_k$, where $C_k = \bot$ is an empty clause (contradiction) and for all i, C_i is either a clause of ϕ or may be obtained from C_j and C_l with $j, l < i$ by the resolution rule. The resolution rule allows to derive a clause $(A \vee B)$ from $x \vee A$ and $\neg x \vee B$. The size of the resolution proof is the number of clauses in it. $S(\phi)$ is the size of the minimal resolution proof of CNF formula ϕ. The width of the resolution proof is the maximal number of literals in the clause from the proof.

It is easy to see that if ϕ has a resolution proof, then ϕ is unsatisfiable. Indeed if an assignment satisfies two clauses then it satisfies the result of the resolution rule. If formula ϕ is satisfiable, then all clauses from its resolution proof are also satisfiable, but empty clause can not be satisfied, we get the contradiction. It is known that Resolution is complete: if ϕ is unsatisfiable, then there is a resolution proof of ϕ.

A treelike resolution proof of formula ϕ is a binary tree with vertices labeled by clauses. Leaves are labeled with clauses of ϕ and internal vertices are labeled by the result of resolution rule of clauses in their children. The root of the tree is labeled by the empty clause. We denote the size of the minimal treelike resolution proof of ϕ as $S_T(\phi)$. A depth of a tree is a length of the maximal path from the root to a leaf. *A depth of a resolution proof* of formula ϕ is the depth of the minimal treelike resolution refutation of ϕ. We denote it as $d(\phi)$.

A regular resolution proof is a directed acyclic graph with one source and several sinks with vertices labeled by clauses. Every internal vertex has outdegree 2. Sinks are labeled by clauses of ϕ and every internal vertex is labeled by the result of resolution rule of clauses in immediately following vertices. The source of the graph is labeled by the empty clause. On every path from the source to a sink resolution rules are applied by distinct variables. We denote the size of the minimal regular resolution proof of ϕ as $S_R(\phi)$.

A decision tree for an unsatisfiable formula ϕ is a binary tree such that its leaves are marked by clauses of ϕ, every internal vertex is marked by a variable of formula ϕ $\{x_1, x_2, \ldots, x_n\}$, one of outgoing edges is marked by 1 and the other by 0. For every assignment σ of variables of ϕ, if we start a path in the root of the tree and on every step in the vertex labeled by x we move along an edge that is labeled by value of x in σ, then we finally will reach a leaf that is labeled by a clause of ϕ that is falsified by σ. $DT(\phi)$ denotes the minimal size of a decision tree for

ϕ. *A read-once branching program* for F is defined by a similar way but instead of a tree we have a directed acyclic graph that has one source and outdegree 2 for all internal vertices. On every path from the source to a sink all variables in vertices are distinct. 1-$BP(\phi)$ denotes the minimal size of a read-once branching program for ϕ.

Proposition 2.1. [9] For any unsatisfiable CNF formula ϕ, $S_T(\phi) = DT(\phi)$, $S_R(\phi)$ =1-$BP(\phi)$ and $d(\phi)$ is the minimal depth over the all decision trees for ϕ.

Let ϕ be a CNF formula, x be a variable of ϕ and $c \in \{0, 1\}$. We denote $\phi[x = c]$ the result of substitution of x by c. To get $\phi[x = c]$ from ϕ we delete all clauses that are satisfied by $x = c$ and delete occurrence of x on all other clauses.

Lemma 2.1. 1) For every unsatisfiable formula ϕ if x is a variable of ϕ, then for all $c \in \{0, 1\}$ the following holds: $S_R(\phi[x = c]) \leq S_R(\phi)$, $S_T(\phi[x = c]) \leq S_T(\phi)$ and $d(\phi[x = c]) \leq d(\phi)$. 2) $S_T(\phi) \leq S_T(\phi[x = 1]) + S_T(\phi[x = 0]) + 1$ and $d(\phi) \leq \max\{d(\phi[x = 1]), d(\phi[x = 0])\} + 1$.

Proof. 1. It is sufficient to prove that a substitution of a variable can't increase the size and the depth of a decision tree or a read-once branching program. Indeed if we have a decision tree for ϕ with vertex v labeled by x, then we remove a subtree corresponding an edge that is labeled by $1 - c$ and also join v with the remain child. We repeat it for all such v. We also should make a substitution $x = c$ to clauses in leaves. In case of read-once branching program the proof is the same.

2. A decision tree for ϕ can be easily constructed from decision trees for $\phi[x = 0]$ and $\phi[x = 1]$: we add new root w labeled by x and add edge labeled by 0 connected w and the root of the decision tree for $\phi[x = 0]$ and edge labeled by 1 connected w and the roof of the decision tree for $\phi[x = 1]$. We also change clauses in leaves by their preimages before substitution.

□

We consider the following game: we have two players Interrogator and Witness. They are given an unsatisfiable CNF formula ϕ. On each step Interrogator chooses a variable of formula ϕ and Witness gives a value for this variable. The game ends if the current substitution refutes some clause of ϕ. For every move Witness earns a coin. The goal of Witness is to earn the maximum number of coins and the goal of Interrogator is to minimize the number of coins earned by Witness.

Lemma 2.2 ([14])**.** If for some unsatisfiable formula ϕ there exists a strategy for Witness that allows Witness to earn t coins, then $d(\phi) \geq t$.

Proof. Assume that $d(\phi) < t$, then by Proposition 2.1 there exists a decision tree T for ϕ with depth less than t. We describe a strategy for Interrogator that guarantees that Witness earns less than t coins; the latter contradicts the statement of the lemma. Interrogator plays according the tree T: he put a pebble on the root of the tree and on each step Interrogator requests the value of a variable in the vertex with the pebble. After the answer of Witness Interrogator moves the pebble along an edge that is labeled by the answer of Witness. The game ends after less than t steps since the depth of T is less than d. We get a contradiction, hence $d(\phi) \geq t$. □

Lemma 2.3 ([14])**.** If for some unsatisfiable formula ϕ there exists such strategy of Interrogator that guarantees that Witness earns at most t coins, then $d(\phi) \leq t$.

Proof. A strategy of Interrogator is actually a decision tree for ϕ. And if Witness earns at most t coins, then the depth of the decision tree is at most t. □

Now consider another but similar game, in this game we have Prover instead of Interrogator and Delayer instead of Witness. Prover chooses the variable and Delayer may return a value from $\{0,1\}$ or return $*$. If Delayer returns $*$, then Prover chooses a value by himself. Now Delayer earns coins only for stars.

Lemma 2.4 ([13])**.** If for some unsatisfiable formula ϕ there exists a strategy for Delayer that allows Delayer to earn t coins, then $S_T(\phi) \geq 2^t$.

Proof. Consider some decision tree T for ϕ. We construct probabilistic distribution on the leaves of T that corresponds to Delayer's strategy and the following randomized strategy of Prover. Prover uses the tree T, initially it asks the question for variable in the root, if Delayer moves $*$, Prover chooses a value at random with equal probabilities and go to the next vertex along an edge labeled with the chosen value. By the statement of the Lemma the probability that the game will finish in every particular leaf is at most 2^{-t}. Since with probability 1 the game will finish in a leaf, the number of leafs of T is at least 2^t. □

Ler ϕ be a CNF formula. A xorification of ϕ is a formula $\phi^{\oplus}$, that is obtained from ϕ by substitutions of x by $x' \oplus x''$ for all variables x from ϕ, here x' and x'' are new variables that have no occurrences in ϕ and this new variables are distinct for all x.

Lemma 2.5 ([14])**.** $S_T(\phi^{\oplus}) \geq 2^{d(\phi)}$ for every unsatisfiable ϕ.

Proof. Lemma 2.3 implies that Witness has a strategy that guarantees him to earn at least $d(\phi)$ coins.

Now we describe the strategy for Delayer in Prover-Delayer game for the formula $\phi^{\oplus}$. Delayer uses the strategy of Witness from the first game; he simulates Interrogator using moves of Prover. Assume that for all variables x xorification substitute $x' \oplus x''$ instead of x. Consider the following cases: 1) Prover asks for the value of x' and did not ask for the value of x'' before this step. In this case Delayer answers $*$. 2) Prover asks for x' and asked for x'' before. The variable x'' has a value $c \in \{0,1\}$. In this case Delayer simulates the step of Interrogator that asks for x. If Witness reports a value b, then Delayer returns $b \oplus c$.

Note that Prover-Delayer game can not finish before Interrogator-Witness game finishes since every clause E of $\phi^{\oplus}$ is a clause of $C^{\oplus}$, there C is a clause of ϕ. If E is refuted in Prover-Delayer game, then $C^{\oplus}$ is also refuted, hence all copies of variables from C have value, by the construction C should be refuted in Interrogator-Witness game.

For every step Witness earns a coin, Delayer earns a coin then Prover asks for the value of the first copy of the variable, hence every coin of Witness corresponds to a coin of Delayer. Thus Delayer earns at least $d(\phi)$ coins, therefore $S_T(\phi) \geq 2^{d(\phi)}$ by Lemma 2.4. □

Theorem 2.1 (Theorem 4.6, [14])**.** There is a family of 3-CNF formulas H_n such that $d(H_n) = \Omega(n/\log n)$, $S_T(H_n) = O(n)$ and H_n has a regular resolution proof of size $O(n)$ and width 3.

Formally Theorem 4.6 from [14] does not claim that the resolution proof of H_n with width 3 is regular, but in fact the resolution proof is a unit-clause propagation and therefore it is regular.

3 Results

3.1 Improved separation of regular and treelike resolution

Lemma 3.1. Let an unsatisfiable CNF formula ϕ have a regular resolution proof of size N and width s. Then $\phi^{\oplus}$ has a regular resolution proof of size at most $3 \cdot 2^s N$.

Proof. Let $C_1, C_2, \ldots, C_N$ be a regular resolution proof of ϕ.

For every literal ℓ we denote the CNF representation of $\ell^{\oplus}$ as $\ell^{(0)} \wedge \ell^{(1)}$, where $\ell^{(0)}$ and $\ell^{(1)}$ are clauses. Consider a clause $C = \ell_1 \vee \ell_2 \vee \cdots \vee \ell_t$ that consists of t literals, the formula $C^{\oplus}$ has the following CNF representation: $C^{\oplus} = \bigwedge_{w \in \{0,1\}^t} \ell_1^{(w_1)} \vee \ell_2^{(w_2)} \vee \cdots \vee \ell_t^{(w_t)}$. We will show that the sequence of clauses $C_1^{\oplus}, C_2^{\oplus}, \ldots, C_N^{\oplus}$ may be extended to a regular resolution proof of $\phi^{\oplus}$.

Note that for every literal ℓ we may derive the empty clause from $(\neg\ell)^{(0)} \wedge (\neg\ell)^{(1)} \wedge \ell^{(0)} \wedge \ell^{(1)}$ using 3 applications of resolution rules (and this derivation is regular). Similarly for any two clauses D_1 and D_2 there is a regular derivation of $D_1 \vee D_2$ from $D_1 \vee (\neg\ell)^{(0)}$, $D_1 \vee (\neg\ell)^{(1)}$, $D_2 \vee \ell^{(0)}$ and $D_2 \vee \ell^{(1)}$ that consists of three applications of resolution rule by variables of $\ell^{\oplus}$. Therefore if C_i is a result of resolution rule applied to C_j and C_k, then every clause from $C_i^{\oplus}$ may be obtained from two clauses of $C_j^{\oplus}$ and two clauses of $C_k^{\oplus}$ by the derivation of size 3. Therefore $C_1^{\oplus}, C_2^{\oplus}, \ldots, C_N^{\oplus}$ may be extended to a resolution proof of $\phi^{\oplus}$ by adding at most two new clauses for every clause. The resulting proof is regular as the initial proof of ϕ was regular and every application of the resolution rule by a new variable corresponds to the application of the resolution rule by the old variable in the initial proof. For all i the formula $C_i^{\oplus}$ contains at most 2^s clauses since C_i has at most s literals. Hence the size of the resulting proof is at most $3 \cdot 2^s N$. □

Theorem 3.1. There exists a family of 6-CNF formulas H'_n that has $O(n)$ variables, $S_R(H'_n) = O(n)$ and $S_T(H_n) = 2^{\Omega(n/\log n)}$.

Proof. Let H_n be a family of 3-CNF formulas from Theorem 2.1 [14] such that $d(H_n) = \Omega(n/\log n)$, $S_T(H_n) = O(n)$ and H_n has a regular resolution proof of size $O(n)$ and width 3. Let $H'_n = H_n^{\oplus}$. By Lemma 2.5 $S_T(H'_n) = 2^{\Omega(n/\log n)}$ and by Lemma 3.1 $S_R(H'_n) = O(n)$. □

3.2 Simultaneous separation of $S_T(\phi), S_R(\phi)$ and $2^{d(\phi)}$

Consider the following formulas that are based on a cellular rectangle $n \times m$. For every cell (i, j) of the rectangle $n \times m$ we have a propositional variable $x_{i,j}$. We assume that the leftmost lowermost cell has coordinates $(1, 1)$ and the rightmost topmost cell has coordinates (n, m). We denote $r(x_{i,j}) = x_{i+1,j}$ if $i < n$ and $r(x_{i,j}) = 0$ otherwise; and $u(x_{i,j}) = x_{i,j+1}$, if $i < m$ and $u(x_{i,j}) = 0$ otherwise. $r(x_{i,j})$ is the right neighbour of $x_{i,j}$ and $u(x_{i,j})$ is the upper neighbour of $x_{i,j}$. By a path in the rectangle $n \times m$ we mean a sequence of cells such that the next cell is either the right or the top neighbour of the previous one.

A formula $\mathrm{Peb}_{n\times m}$ is the conjunction of clauses $\neg x_{i,j} \vee u(x_{i,j}) \vee r(x_{i,j})$ for all $i \in [n], j \in [m]$ and clause $x_{1,1}$. Formula $\mathrm{Peb}_{n\times m}$ states that if the value of some variable is 1, then its upper or right neighbour also should have the value 1 and that the leftmost and lowermost variable has value 1. This formula is unsatisfiable since otherwise there should be a path with values 1; it starts in $x_{1,1}$ and ends in $x_{n,m}$. Note that $x_{n,m}$ has no upper or right neighbours (the formula contains clause $\neg x_{n,m}$) and therefore it can not have value 1.

Proposition 3.1. $S(\mathrm{Peb}_{n\times m}) \geq mn + 2$.

Proof. It is sufficient to show that all clauses of $\mathrm{Peb}_{n\times m}$ must be used in the resolution refutation of $S(\mathrm{Peb}_{n\times m})$. Indeed there are exactly $nm + 1$ clauses and the empty clause is always in the resolution refutation. We show that the removing of any clause makes formula satisfiable. If we remove clause $x_{1,1}$, then all zeros assignment satisfies all other clauses. If we remove a clause that corresponds to (i, j), then the assignment that assigns 1 to cells $x_{1,1}, x_{1,2}, \ldots, x_{1,j}, x_{2,j}, \ldots, x_{i,j}$ and zeros to other cells satisfies all clauses. □

Proposition 3.2. $S_T(\mathrm{Peb}_{n\times m}) \leq nm + 2$.

Proof. We consequently apply resolution rule to clauses in cells: we start from the leftmost and lowermost cell $(1, 1)$ and go through all cells by diagonals: $(1, 1)$, $(2, 1)$, $(1, 2)$, $(3, 1)$, $(2, 2)$, $(1, 3)$,.... We maintain one clause and on every step we change it by the result of the resolution rule with the clause in the current cell. Finally we get a clause $\neg x_{1,1}$ and then we resolve it with $x_{1,1}$ and get the empty clause. □

Corollary 3.1. $S(\mathrm{Peb}_{n\times m}) = S_R(\mathrm{Peb}_{n\times m}) = S_T(\mathrm{Peb}_{n\times m}) = mn + 2$

Proof. By Proposition 3.1 we have $mn+2 \leq S(\mathrm{Peb}_{n\times m})$ and by Proposition 3.2, $S_T(\mathrm{Peb}_{n\times m}) \leq mn + 2$; the inequality $S(\phi) \leq S_R(\phi) \leq S_T(\phi)$ holds for all unsatisfiable ϕ. □

In what follows we assume that $m = n$ and we consider formulas based on squares.

Lemma 3.2. $d(\mathrm{Peb}_{n\times n}) \geq n/2$.

Proof. In order to use Lemma 2.2 we describe a strategy for Witness in the Interrogator-Witness game. Witness maintains the following invariant:

- Witness maintains a path P from $x_{1,1}$ to $x_{r,r}$, where $r \in [n]$ (recall that we consider paths where the next cell is the upper or the right neighbour of the previous).
- If Interrogator asked the values of variables from P, Witness gave answer 1, for all other variables Witness gave answer 0
- Interrogator did not make requests to cells $x_{r,j}$ and $x_{j,r}$ for $r > i$.

Initially P consist of one cell $x_{1,1}$, i.e. $r = 1$. If Interrogator asks a value of cell from P, then Witness gives answer 1, otherwise Witness gives answer 0. Witness changes P if Interrogator makes query to the cell $x_{r,j}$ or $x_{j,r}$ for $j > r$. Assume that Interrogator asks a value of $x_{r,j}$ (the second case is similar), where $j > r$. In that case Witness finds the minimal $r' > r$ such that Interrogator did not make requests to all cells $x_{r',j}$ and $x_{j,r'}$ for all $j \geq r$. If there are no such r', then Witness gives up. If there is such r', then Witness prolongs the path P by the following way: $x_{r+1,r}, \ldots, x_{r',r}, x_{r',r+1}, \ldots x_{r',r'}$ and changes $r := r'$. We note that by this moment there were no Interrogator's request to new cells of P.

A cross with center k is the set of all cells $x_{k,j}$ and $x_{j,k}$ for all $j \in [n]$. Note that if in some moment P ends in $x_{r,r}$, then Interrogator made requests for every cross with centers in $1, 2, \ldots r - 1$. The game ends in two situations: 1) Witness gave up; in this case Interrogator made requests for all crosses and thus Witness have already earned at least $\frac{n}{2}$ coins, since every cell is in at most two crosses. 2) P ends in $x_{n,n}$ and Interrogator asks the value of $x_{n,n}$. In this case Witness already earned at least $\frac{n-1}{2}$ coins for crosses with centers $1, 2, \ldots, n - 1$ and will earn 1 coin for the last step. Therefore Witness earns at least $\frac{n}{2}$ coin, thus $d(\mathrm{Peb}_{n\times n}) \geq n/2$ by Lemma 2.2. □

Corollary 3.2. $S_T(\mathrm{Peb}^{\oplus}_{n\times n}) \geq 2^{n/2}$.

Proof. Follows from Lemma 2.5. □

Corollary 3.3. $d(\mathrm{Peb}^{\oplus}_{n\times n}) \geq n/2$.

Proof. Follows from Corollary 3.2 and inequality $S_T(\phi) \leq 2^{d(\phi)}$. □

Proposition 3.3. 1) $d(\mathrm{Peb}_{n\times n}) \leq 3n$; 2) $d(\mathrm{Peb}^{\oplus}_{n\times n}) \leq 6n$.

Proof. 1) In order to use Lemma 2.3 we have to describe the strategy of the Interrogator that guarantee that Witness earns at most $3n$ coins. The idea is the following: if we make several requests to variables of unsatisfiable formula ϕ such that after substitution the formula will split into two parts ϕ_1 and ϕ_2 that have no common variables, then at least one of ϕ_1 and ϕ_2 is unsatisfiable and Interrogator may makes all further requests only to one of them. Two clauses of $\mathrm{Peb}_{n\times n}$ has common variable if they correspond to adjacent cells. At first Interrogator makes n requests to cells from the column number $\lceil \frac{n}{2} \rceil$. After it we choose unsatisfiable rectangle of size at most $n \times \lfloor n/2 \rfloor$. This rectangle may be splitted on two of approximate equal size by $\lfloor n/2 \rfloor$ requests. So Interrogator makes at most $\frac{3n}{2}$ requests and reduce the rectangle $n \times n$ to rectangle with sides at most $\frac{n}{2}$. We continue decreasing the size of the rectangle in 2 times. In some moment one of the clauses would be unsatisfiable. Totally Interrogator makes at most $\frac{3n}{2} + \frac{3n}{4} + \frac{3n}{8} + \cdots \leq 3n$ request; therefore Witness earns at most $3n$ coins. 2) The proof is the same as in previous case. Now every cell corresponds to two variables and hence Interrogator need to make two times more requests. □

Proposition 3.4. $S_R(\mathrm{Peb}^{\oplus}_{n\times n}) = O(n^2)$.

Proof. We will show that $\mathrm{Peb}_{n\times n}$ has a regular resolution proof of size $O(n^2)$ and width 3. We go through all cells from the rightmost uppermost cell by diagonals and derive $\neg x_{i,j}$ initially for $i+j = 2n$, then for $i+j = 2n-1$ etc. Finally we get $\neg x_{1,1}$ and using $x_{1,1}$ derive the empty clause. The proof is regular since it is a unit-clause propagation.

Lemma 3.1 implies that $\mathrm{Peb}^{\oplus}_{n\times n}$ has a regular proof of size $O(n^2)$.

□

We define formulas $\mathrm{Peb}^{k,\oplus}_{n\times n}$, that can be obtained from $\mathrm{Peb}_{n\times n}$ by substitution of $\oplus$ of two new variables for all variables from leftmost downmost square $k \times k$ (i.e. variables $x_{i,j}$, where $i, j \leq k$).

Theorem 3.2. The following holds for all $k \in [n]$: 1) $S_R(\mathrm{Peb}^{k,\oplus}_{n\times n}) = \Theta(n^2)$; 2) $d(\mathrm{Peb}^{k,\oplus}_{n\times n}) = \Theta(n)$; 3) $\max\{n^2, 2^{k/2}\} \leq S_T(\mathrm{Peb}^{k,\oplus}_{n\times n}) \leq 2^{6k} + 2k(n^2+1)$;

Proof. 1) The upper bound follows from Proposition 3.4, since the formula $\mathrm{Peb}^{k,\oplus}_{n\times n}$ can be obtained from $\mathrm{Peb}^{\oplus}_{n\times n}$ by the substitution of zeros instead of several variables. The lower bound follows from Proposition 3.1, since $\mathrm{Peb}_{n\times n}$ can be obtained from $\mathrm{Peb}^{k,\oplus}_{n\times n}$ by the substitution of zeros instead of several variables.

2) The upper bound follows from Proposition 3.3 and the fact that $\mathrm{Peb}^{k,\oplus}_{n\times n}$ can be obtained from $\mathrm{Peb}^{\oplus}_{n\times n}$ by the substitution of zeros instead of several variables. The lower bound follows from Lemma 3.2 and the fact that $\mathrm{Peb}_{n\times n}$ can be obtained from $\mathrm{Peb}^{k,\oplus}_{n\times n}$ by the substitution of zeros instead of several variables.

3) The lower bound $2^{k/2}$ follows from the fact that after substitution $x_{i,j} = 0$ for all $i > k$ and $j > k$ the formula $\mathrm{Peb}^{k,\oplus}_{n\times n}$ becomes $\mathrm{Peb}^{\oplus}_{k\times k}$, the complexity of the latter is at least $2^{k/2}$ by Corollary 3.2. Lower bound n^2 follows from Proposition 3.1.

The upper bound. If $k = n$, then $\mathrm{Peb}^{k,\oplus}_{n\times n} = \mathrm{Peb}^{\oplus}_{n\times n}$. Then by Proposition 3.3, $d(\mathrm{Peb}^{n,\oplus}_{n\times n}) \leq$

$6n$, therefore $S_T(\mathrm{Peb}^{n,\oplus}_{n\times n}) \leq 2^{6n}$. Let $k < n$. Consider formulas

$$\phi_1 = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 1], \phi_2 = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, x_{k+1,2} = 1],$$
$$\phi_3 = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, x_{k+1,2} = 0, x_{k+1,3} = 1], \ldots,$$
$$\phi_k = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, \ldots, x_{k+1,k-1} = 0, x_{k+1,k} = 1],$$
$$\phi_{k+1} = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, \ldots, x_{k+1,k} = 0, x_{1,k+1} = 1],$$
$$\phi_{k+2} = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, \ldots, x_{k+1,k} = 0, x_{1,k+1} = 0, x_{2,k+1} = 1], \ldots,$$
$$\phi_{2k} = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, \ldots, x_{k+1,k} = 0, x_{1,k+1} = 0 \ldots, x_{k-1,k+1} = 0, x_{k,k+1} = 1],$$
$$\phi_0 = \mathrm{Peb}^{k,\oplus}_{n\times n}[x_{k+1,1} = 0, \ldots, x_{k+1,k} = 0, x_{1,k+1} = 0, \ldots, x_{k,k+1} = 0].$$

Note that all formulas ϕ_i for $i \in [k]$ contains a subformula that is isomorphic to $\mathrm{Peb}_{(n-k)\times(n-i+1)}$; formulas ϕ_i for $k+1 \leq i \leq 2k$ contains a subformula that is isomorphic to $\mathrm{Peb}_{(n-i+k+1)\times(n-k)}$. Hence by Proposition 3.2 formulas ϕ_i for $i \in [2k]$ has treelike resolution proof of size at most n^2. Formula ϕ_0 has subformula that is isomorphic to $\mathrm{Peb}^{k,\oplus}_{k\times k}$, in the case $n = k$ we show that this subformula has resolution proof of size at most 2^{6k}. By Lemma 2.1 for all ϕ, $S_T(\phi) \leq S_T(\phi[x_1 = 0]) + S_T(\phi[x_1 = 1]) + 1$. We apply $2k$ times Lemma 2.1 to formulas $\phi_0, \phi_{2k}, \phi_{2k-1}, \ldots, \phi_i$ and get a treelike resolution proof of $\mathrm{Peb}^{k,\oplus}_{n\times n}$ of size at most $2^{6k} + 2kn^2 + 2k$. □

Acknowledgements. The author is grateful to Anna Malova and Alexander Knop for fruitful discussions and to Dmitry Sokolov and Mikhail Slabodkin for useful comments.

References

[1] Michael Alekhnovich, Jan Johannsen, Toniann Pitassi, and Alasdair Urquhart. An exponential separation between regular and general resolution. *THEORY OF COMPUTING*, 3:81–102, 2007.

[2] A. Atserias, J. K. Fichte, and M. Thurley. Clause-learning algorithms with many restarts and bounded-width resolution. *Journal of Artificial Intelligence Research*, 40:353–373, 2011.

[3] Eli Ben-Sasson, Russell Impagliazzo, and Avi Wigderson. Near optimal separation of treelike and general resolution. *Combinatorica*, 24:585–604, 2003.

[4] Maria Luisa Bonet, Juan Luis Esteban, Nicola Galesi, and Jan Johannsen. On the relative complexity of resolution refinements and cutting planes proof systems. *SIAM J. Comput.*, 30(5):1462–1484, May 2000.

[5] Stephen A. Cook. An observation on time-storage trade off. *Journal of Computer and System Sciences*, 9:308–317, 1974.

[6] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5:394–397, 1962.

[7] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7:201–215, 1960.

[8] Andreas Goerdt. Regular resolution versus unrestricted resolution. *SIAM J. Comput.*, 22:661–683, 1993.

[9] Stasys Jukna. *Boolean function complexity: advances and frontiers*, volume 27. Springer Science & Business Media, 2012.

[10] Jan Krajicek. *Bounded Arithmetic, Propositional Logic and Complexity Theory*, volume 60 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, 1995.

[11] Wolfgang J. Paul, Robert Endre Tarjan, and James R. Celoni. Space bounds for a game on graphs. *Mathematical Systems Theory*, 11:85, 1977.

[12] Knot Pipatsrisawat and Adnan Darwiche. On the power of clause-learning sat solvers as resolution engines. *Artificial Intelligence*, 175:512–525, 2011.

[13] Pavel Pudlák and Russell Impagliazzo. A lower bound for DLL algorithms for k-sat (preliminary version). In *Proceedings of the Eleventh Annual ACM-SIAM Symposium on Discrete Algorithms, January 9-11, 2000, San Francisco, CA, USA.*, pages 128–136, 2000.

[14] Alasdair Urquhart. The depth of resolution proofs. *Studia Logica*, 99(1-3):249–364, 2011.

[15] Alasdair Urquhart. A near-optimal separation of regular and general resolution. *SIAM J. Comput.*, 40:107–121, 2011.