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 ϕ, uses a procedure $\mathbf{A}$ to choose a variable x, uses a procedure $\mathbf{B}$ 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 [1], [2]. In case of satisfiable formulas we have no hope to prove superpolynomial lower bound since if P = NP, then procedure $\mathbf{B}$ may always choose the correct value of the variable according to some satisfying assignment. Formulas that encode unsatisfiable systems of linear equations are hard for resolution and hence for DPLL [2], [3]. Systems of linear equations are also hard satisfiable examples for myopic and drunken DPLL algorithms [4], [5]. Hard examples for myopic algorithms with a cut heuristic are also based on linear systems [6]. In paper [7] we show that a splitting by linear combinations helps to solve explicitly encoded linear systems over $\mathbb{F}_2$ in polynomial time.

For every CNF formula ϕ we denote by $\phi^{\oplus}$ a CNF formula obtained from ϕ by substituting $x_1 \oplus x_2$ for each variable x. Urquhart shows that for unsatisfiable ϕ the running time of any DPLL algorithm on $\phi^{\oplus}$ is at least $2^{d(\phi)}$, where $d(\phi)$ is the minimal depth of the recursion tree of DPLL algorithms running on the input ϕ [8]. Urquhart also gives an example of Pebbling contradictions $Peb(G_n)$ such that $d(Peb(G_n)) = \Omega(n/\log n)$ and there is a DPLL algorithm that solves $Peb(G_n)$ in $O(n)$ steps. Thus $Peb^{\oplus}(G_n)$ is one more example that is hard for DPLL algorithms but easy for DPLL with splitting by linear combinations.

In paper [7] we prove an exponential lower bound on the size of a splitting tree by linear combinations for 2-fold Tseitin formulas that can be obtained from ordinary Tseitin formulas by substituting every variable by the conjunction of two new variables. We also give an elementary proof of the lower bound $2^{\frac{n-1}{2}}$ on the size of linear splitting trees of formulas PHP^m_n that encode the pigeonhole principle.

We consider the extension of the resolution proof system that operates with disjunctions of linear equalities. A system Res-Lin contains the weakening rule and the resolution rule. We also consider a system Sem-Lin that is a semantic version of Res-Lin; Sem-Lin contains semantic implication rule with two premises instead of the resolution rule. We prove that this two systems are polynomially equivalent and they are implication complete. We also show that tree-like versions of Res-Lin and Sem-Lin are equivalent to linear splitting trees; the latter implies that our lower bounds hold for tree-like Res-Lin and Sem-Lin.

Further research Let's consider space complexity of Res-Lin proofs similarly to the Resolution [9]. We assume that a proof is realized in the working memory. And there are the following basic operations: 1) To download a clause of the formula to the memory; 2) To remove a clause from the memory; 2) To deduce a clause form clauses in the memory using inference rules and add it to the memory. A clause space of a proof is the maximum number of clauses in the memory. We denote a clause space of π as $CSpace(\pi)$ and the number of operations in π as $Size(\pi)$.

The first problem is the proof of trade-off between clause space and the size of proof for any formula ϕ. The planned result is the following: $R(Search_\phi) \leq CSpace(\pi) \log(Size(\pi))$, where R is the communication complexity of problem $Search_\phi$. It means that where exists formula ϕ such that $n^{1/3-\epsilon} \leq CSpace(\pi) \log(Size(\pi))$.

The second problem is proof unconditioned lower bound on space complexity by using Atserias-Dalmau games [10]. The main goal to prove following result: there exists formula ϕ such that for any proof π in $Res - Lin$ $CSpace(\pi) = \Omega(n^\epsilon)$.

Список литературы

[1] A. Urquhart. Hard examples for resolution. *JACM*, 34(1):209–219, 1987.

[2] 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.

[3] E. Ben-Sasson and A. Wigderson. Short proofs are narrow — resolution made simple. *Journal of ACM*, 48(2):149–169, 2001.

[4] 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.

[5] Dmitry Itsykson. Lower bound on average-case complexity of inversion of goldreich's function by drunken backtracking algorithms. *Theory Comput. Syst.*, 54(2):261–276, 2014.

[6] 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.

[7] Dmitry Itsykson and Dmitry Sokolov. Lower bounds for splittings by linear combinations. In *Mathematical Foundations of Computer Science 2014 - 39th International Symposium, MFCS 2014, Budapest, Hungary, August 25-29, 2014. Proceedings, Part II*, pages 372–383, 2014.

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

[9] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov, and Avi Wigderson. Space complexity in propositional calculus. *SIAM J. Comput.*, 31(4):1184–1211, April 2002.

[10] Albert Atserias and Víctor Dalmau. A combinatorial characterization of resolution width. *Journal of Computer and System Sciences*, 74(3):323 – 334, 2008. Computational Complexity 2003.