DOI: 10.17516/1997-1397-2021-14-5-667-671
УДК 512.54

Satisfiability in Boolean Logic (SAT problem) is polynomial

Vladimir V. Rybakov*
Siberian Federal University
Krasnoyarsk, Russian Federation
A. P. Ershov Institute of Informatics Systems
Novosibirsk, Russian Federation

Received 10.07.2021, received in revised form 10.08.2021, accepted 21.08.2021

Abstract. We find a polynomial algorithm to solve SAT problem in Boolean Logic.

Keywords: Boolean Logic, Satisfiability Problem, SAT algorithm.

Citation: V.V. Rybakov, Satisfiability in Boolean Logic (SAT problem) is polynomial, J. Sib. Fed. Univ. Math. Phys., 2021, 14(5), 667–671. DOI: 10.17516/1997-1397-2021-14-5-667-671.

Introduction

The satisfiability problem (SAT) in Boolean propositional logic is the question to determine if any given formula F is satisfiable (i.e. if there is a substitution of literals TUE-$FALSE$ instead the propositional letters from given formula F making the formula $TRUE$). Extended SAT problem is to find a such substitution if the one exists. SAT is an NP-complete problem, and is one of the most intensively studied problems.

As well known SAT was the first known NP-complete problem, that was proved by S.Cook at the University of Toronto in 1971 (cf. [1]) and also independently by L.Levin in 1973 (cf. [2]). Remarkably that before these results, the idea, the concept, of an NP-complete problem did not even exist, so was totally out of consideration. .

That generated a very active area in complexity theory; since the SAT problem is NP-complete, and only algorithms with exponential worst-case complexity are until now known for it, better algorithms for SAT where in grate demand. In particular researchers looked for efficient and scalable algorithms for SAT for formulas in restricted form; and during the 2000s algorithms making dramatic advances in our ability to automatically solve problem was developed (cf. [3, 4, 10–12]). In this paper we will prove that SAT may be solved in polynomial time.

1. Proof, Deterministic Algorithm with Random Any Choice

We first need to restrict the amount of necessary formulas in our considerations. We will do this restriction by Theorem 1 placed below. Actually this result was known for long ago, for example the author introduced reduced normal forms for inference rules in [5, 6] where he solved Friedman problem about recognition inference rules for intuitionistic propositional logic. This technique was efficiently applied in [5–9] for study inference rules and unification. The point here is that the premises of such rules are exactly the normal reduced forms for just formulas. These approach also possibly was observed even earlier when researchers used reduction formulas for

*Vladimir_Rybakov@mail.ru
© Siberian Federal University. All rights reserved

3-Sat problem and relative subsequent research (cf. [3,4,10–12]). I am not sure about history and priority not being very expert in the SAT area. Though it turned out that Theorem 1 is also very useful for positive solution SAT problem and we present it now.

Definition. *We say that Boolean formula F has reduced normal form if*

$$F = A_1 \wedge A_2 \wedge \cdots \wedge A_k,$$

where

$$A_j := B_{j_1} \vee B_{j_2} \vee B_{j_3},$$

and $B_{j_i} \in \{p, \neg p\}$, $p \in Prop$ and $Prop$ is a set of letters.

In the sequel we may consider also A_j containing less then 3 disjunct members. Simply for convenience and simplicity in notation we will always refer to 3 disjunct members, thinking that it might be less.

Theorem 1. *There is a polynomial algorithm constructing by any given boolean formula G a formula F in reduced normal form which has the following properties. (1) F has all variables of G and some more in amount not bigger then the length of G. (2) F is equivalent to G w.r.t satisfiability. (3) Any substitution σ for F satisfying G acting at only variables of G satisfies G, (and vice versa any substitution satisfying G may be computably extended on additional variables of F satisfying it).*

Proof. It is a simple statement; as much as I remember I myself proved it first time in my works for constructing reduced normal forms for inference rules in my research to resolve Friedman problem about recognizing admissible rules in intuitionistic logic cf. [6], 1984. A short draft of the proof is as follows. In fact, we simply shall specify the general algorithm described already several times in [5–9] to the language of our logic.

So, let we start. If $\varphi = \alpha \circ \beta$, where $\circ$ is a binary logical operation and both formulas α and β are not simply variables or unary logical operations applied to variables (which both we call final formulas), take two new variables x_α and x_β and the formula

$$\mathbf{f_1} := (x_\alpha \circ x_\beta) \wedge (x_\alpha \equiv \alpha) \wedge (x_\beta \equiv \beta).$$

If one from formulas α or β is final and another one not, we apply this transformation to the non-final formula. It is clear that $\mathbf{f}$ and $\mathbf{f_1}$ are equivalent w.r.t. satisfiability.

If $\varphi = \neg\alpha$ and α is not a variable, take a new variable x_α and the formula

$$\mathbf{f_1} := \neg x_\alpha \wedge (x_\alpha \equiv \alpha).$$

Again $\mathbf{f}$ and $\mathbf{f_1}$ are equivalent w.r.t. satisfiability. We continue this transformation over the resulting formulas

$$\bigwedge_{j \in J_1} \gamma_j \wedge \bigwedge_{i \in I_1} x_{\alpha_i} \equiv \alpha_i$$

until all formulas α_i and γ_j in the formula above will be either atomic formulas, i.e. logical operations applied to variables, or variables.

Evidently this transformation is *polynomial*. Further, we transform the formulas using $\equiv$ in the ones using only disjunctions and conjunctions and negations. After that we obtain formula in form as required for reduced normal forms. The only point is that the conjuncts may continue less that 3 disjunct formulas, we then may double some until 3 members. As the result we get the final formula $\mathbf{f_2}$ which evidently has all required properties. Q.E.D.

Now we turn to SAT problem. In the sequel a literal is either a propositional letter or a letter with applied negation ($\neg p$ for p). Below we will always refer to only 3 disjuncts in $F = A_1 \wedge A_2 \wedge \cdots \wedge A_k$,, where $A_j := B_{j_1} \vee B_{j_2} \vee B_{j_3}$. But we could admit less disjuncts, simply we keep all 3 for simplicity of notation.

Theorem 2. *If a formula F has reduced normal form then there is a polynomial algorithm verifying its satisfiability and constructing its some unifier if F is satisfiable.*

Proof. Let

$$F = A_1 \wedge A_2 \wedge \cdots \wedge A_k,$$

where

$$A_j := B_{j_1} \vee B_{j_2} \vee B_{j_3},$$

and $B_{j_i} \in \{p, \neg p\}, p \in Prop$ where $Prop$ are propositional letters. Assume F has exactly m letters.

It is evident that F is satisfiable iff there is at least one path from A_1 to A_k passing through each A_j via some unique B_{j_i} (in this A_j) not containing contradictory literals along all path.

We will try to construct such a path now. We do it by induction on j in A_j so we do it by induction on k in $A_1 \wedge A_2 \wedge \cdots \wedge A_k$. Let $k = 1$ then we have the path.

Inductive step. Assume that n steps are already done and (1) all sets $Imp(B_{n_i})$ are constructed and any $Imp(B_{n_i})$ contains absolutely all possible literals g (in given m letters) to which we cannot make step from B_{n_i} that is g is any literal contradicting to some lateral in any possible not contradictory paths leading from any disjunct from A_1 to B_{n_i}. (2) The sets $To(B_{n_i})$ contain all B_{n-1_j} which themselves are reached by non-contradictory (no matter which) paths from A_1 and from which we (non-contradictory) moved in B_{n_i}. Note that we do not store (record or memorize) paths themselves - we just fix (record) their existence by marking all B_{n-1_j} from which we made final steps to (B_{n_i}). (That is why we summarize things (steps, actions) while the procedure but do not multiply them.)

Of course we assume all sets $To(B_{1_i})$ to be $\{B_{1_i}\}$ and for $n = 1$ all is ready, and for all i, $Imp(B_{1_i}) = \{\neg B_{1_i}\}$ if B_{1_i} is a letter and $Imp(B_{1_i}) = \{B_{1_i}\}$ otherwise.

Now we turn to the inductive step itself, we try to move to $(n+1)$-th conjunct from A_n, to make $(n+1)$-th step. Assume that n steps are done and we arrived to B_{n_i} (which informally means by not-contradictory path; which in our formalism only means that $To(B_{n_i}) \neq \emptyset$, or $n = 1$) and all $Imp(B_{n_j})$ and $To(B_{n_j})$ are constructed. Consider any B_{n+1_j}. Define immediately

$$Imp(B_{n+1_j}) := [Imp(B_{n_1}) \cap Imp(B_{n_2}) \cap Imp(B_{n_3})] \cup \{\neg B_{n+1_j}\}$$

and

$$To(B_{n+1_j})$$

are unions of all B_{n_l} which do not contradict B_{n+1_j} and where $To(B_{n,l})$ are not empty (which, by the way, means that $B_{n,l}$ are reached by some non-contradictory paths, and, recall the inductive assumption, — sets $Imp(B_{n_l})$ are already successfully constructed).

It is clear that $Imp(B_{n+1_j})$ is the set containing all literals to which we cannot step form B_{n+1_j} further at all, even, in particular, to literals which occurs A_{n+2}. Consider all B_{n+2_l} from level $n+2$; if any of them occurs in $Imp(B_{n+1_j})$ we cross out such B_{n+1_j} from further consideration. And if that indeed holds for all B_{n+1_l} the procedure stops and formula F is not satisfiable.

If we can continue, recall that earlier we put in $To(B_{n+1_j})$ all B_{n_i} from which we arrived to B_{n+1_j}. The step (n+1) is completed.

If we came to some B_{k_j} the formula F is satisfiable otherwise not. And the sets $To(B_{n+1_j})$ give us a satisfying substitution, if we will move from B_{k_j} back to A_1 via sets $To(B_{k_j})$ subsequently using $To(B_{n_j})$ towards A_1. The trick here is that we do not do any choice at all; during moving

we just take any B_{k_j} from $To(B_{n+1_j})$ (and we have at most 3 options for that each choice) and move towards A_1.

The interesting thing here is that we do not do choice at all — we can take subsequently any from any occurring in B_{n_j} moving to the origin A_1. That looks as not deterministic algorithm but in fact it is the one, a good one, since we can take any disjunct from at most 3 possible options and any of them will lead us to success.

The amount of steps in this algorithm is polynomial: the general amount of steps used in our inductive procedure in constructing $Imp(B_{n_j})$ and $To(B_{n+1_j})$ is at most k. And in the inductive step itself from n to $n+1$ for constructing $Imp(B_{n+1_j})$ and $To(B_{n+1_j})$, the amount of steps is at most $3 \times [3 \times (2m)^2]$ (m is the number of letters in formula) for computing intersections $Imp(B_{n_1}) \cap Imp(B_{n_2}) \cap Imp(B_{n_3})]$. Q.E.D.

This research is supported by High Schools of Economics (HSE) Moscow; supported by the Krasnoyarsk Mathematical Center and financed by the Ministry of Science and Higher Education of the Russian Federation (Grant No. 075-02-2020-1534/1).

References

[1] S.Cook, The complexity of theorem proving procedures, Proceedings of the Third Annual CAM Symposium on Theory of Computing, 1971, 151–158.

[2] L.A.Levin, Universal sequential search problems, *Problemy Peredachi Informatsii*, **9**(1973), no. 3, 115–116 (in Russian); Translated into English by B.A.Trakhtenbrot, A survey of Russian approaches to perebor (brute-force searches) algorithms, *Annals of the History of Computin*, **6**(1984), no. 4, 384–400.

[3] B.Selman, D.Mitchell, H.Levesque, Generating Hard Satisfiability Problems, *Artificial Intelligence*, **81**(1996), no. 1, 1729.

[4] T.J.Schaefer, (1978) The complexity of satisfiability problems, Proceedings of the 10th Annual CAM Symposium on Theory of Computing, San Diego, California, 216226.

[5] V.Rybakov, Admissible Rules in Pretabular Modal Logic, *Algebra and Logic*, **20**(1981), 291–307.

[6] V.Rybakov, Criterion for Admissibility for Rules in the Modal System S4 and Intuitionistic Logic, *Algebra and Logic*, **23**(1984), no. 5, 291–307.

[7] V.V.Rybakov, Logical equations and admissible rules of inference with parameters in modal provability logics, *Studia Logica*, **49**(1990), no. 2, 215–239

[8] V.V.Rybakov, Problems of substitution and admissibility in the modal system Grz and in intuitionistic propositional calculus *Annals of Pure and Applied Logic*, **50**(1990), no. 1, 71–106.

[9] V.V.Rybakov, Admissibility of Logical Inference Rules, Vol. 136, 1st Edition, Elsevier, 1997.

[10] F.Massacci, L.Marraro, Logical Cryptanalysis as a SAT Problem, *Journal of Automated Reasoning*, **24**(2000), no. 1, 165203.

[11] Marijn J.H.Heule, Hans van Maaren, Look-Ahead Based SAT Solvers, Handbook of Satisfiability, IOS Press, 2009, 155184.

[12] C.Moore, S.Mertens, The Nature of Computation, Oxford University Press, 2011.

Проблема выполнимости формул в булевой логике (SAT) полиномиальна?

Владимир В. Рыбаков
Сибирский федеральный университет
Красноярск, Российская Федерация
Институт систем информатики им. А. П. Ершова
Новосибирск, Российская Федерация

Аннотация. Находится полиномиальный алгоритм решающий проблему SAT в Булевой логикуе.

Ключевые слова: булева логика, проблема выполнимости, алгоритм SAT.