Relaxed Random Search for Solving K-Satisfiability and its Information Theoretic Interpretation

Amirahmad Nayyeri
Department of Science, Division of Computer Science
Salman Farsi University of Kazerun
Kazerun, Iran

Gholamhossein Dastghaibyfard
Department of Computer Engineering and IT
Shiraz University
Shiraz, Iran

***Abstract*—The problem of finding satisfying assignments for conjunctive normal formula with K literals in each clause, known as K-SAT, has attracted many attentions in the previous three decades. Since it is known as NP-Complete Problem, its effective solution (finding solution within polynomial time) would be of great interest due to its relation with the most well-known open problem in computer science (P=NP Conjecture). Different strategies have been developed to solve this problem but in all of them the complexity is preserved in NP class. In this paper, by considering the recent approach of applying statistical physic methods for analyzing the phase transition in the complexity of algorithms used for solving K-SAT, we try to compute the complexity of using randomized algorithm for finding the solution of K-SAT in more relaxed regions. It is shown how the probability of literal flipping process can change the complexity of algorithm substantially. An information theoretic interpretation of this reduction in time complexity will be argued.**

***Keywords*—Constraint satisfaction problem; K-SAT; threshold phenomena; randomized algorithm; entropy; NP-completeness**

I. Introduction

In computer science, there is an important family of problems known as Constraint Satisfaction problem. In this family we are looking for the values of variables which satisfy the set of constraints simultaneously. Although many of these constraints deals with non-Boolean variables, but they can be reduced to the well-known form of satisfying a canonical form of logical formula called Conjunctive Normal Form (CNF). When each clause in CNF has K literals, this problem is called K-Satisfiability problem or K-SAT.

This problem covers a wide range of different theoretical and applied problems. Scheduling time table problem [1], Planning in Artificial Intelligence [2], validating software models [3], routing field programmable gate arrays [4] and synthesizing consistent network configurations [5] are recognized among these problems.

Furthermore, an important problem of designing digital circuits and their verifications can be reformulated easily into the satisfying of K-SAT formula [6]-[8].

The theoretical reason behind this wide range of application for K-SAT problem was discovered by Stephen Cook [9]. He proved the NP-Complete nature of the K-SAT problem. It means all of NP problems can be reduced to the version of K-SAT problem by using an efficient procedure with polynomial time complexity [10]. Therefore it is not hard to imagine that how much the effective strategy for solving K-SAT would be advantageous, both from theoretical and practical perspectives.

This paper has been organized into four sections. After this primary introduction about K-SAT, different strategies which have been designed to solve k-SAT are reviewed in Section 2. Section 3 focuses on randomized algorithm, in which it is tried to improve the time complexity of algorithm by relaxing the conditions imposed on random walking in the solution space inspired by the recent studies about the typical time complexity of K-SAT problem [11]. Finally in Section 4, concluding remarks and future works will be discussed.

II. Related Works

Classically constraint satisfaction problems are solved by systematic search algorithms. For K-SAT problem, this approach is followed in the DPLL algorithm [12], [13].

In this algorithm after choosing a value for any unassigned variable, the formula is simplified by considering the propagation of chosen value in the formula. Since the constraints are represented in conjunctive normal form, two main equivalence rules in propositional logic about disjunctive phrases are used to deduce the consequences of any variable assignment (1) & (2).

$$expression \vee True \equiv True \quad (1)$$

$$expression \wedge False \equiv False \quad (2)$$

For any K-SAT formula on n Boolean variables, 2^n assignments are possible. This exponentially large state space can be pruned by considering the structural properties of the CNF formula. Sometimes it is better to start the process of assigning value to variables from highly constrained variables and sometimes it is better to start with more relaxed variables.

In the case of applying DPLL algorithm, the process of assignment is started with unit-clause (single literal clause). These unite clauses provide a suitable way to reduce the size of state space by imposing a strict type of restriction on the value of literals in the unite-clauses.

The best scenario happens when deducing the consequences of any unite-clause assignment provide an opportunity for forming another unite-clause. Consider the following formula:

$$\varphi(x_1, x_2, \dots, x_n) = (\overline{x_1}) \wedge (x_1 \vee x_2) \wedge (\overline{x_2} \vee x_5) \wedge \dots \quad (3)$$

Certainly for satisfying φ , x_1 must be assigned 0, considering this value forx_1, the second clause is transformed into a unit clausex_2. Therefore x_2 must be assigned 1 or true. Again considering this value for x_2 the third clause is transformed into unit clause x_5 and the process is continued by choosing proper values for the variables of formula. This condition provides a clear guide for choosing the values for variables without any doubt and reduces the time complexity of the problem.

Unfortunately this consecutive formation of unite clauses happens rarely and cannot be used as a general technique. As a matter of fact the consecutive emergence of unit clauses in the process of deducing the consequences of the variable assignment reduces the branching factor of the search tree.

Studies have shown the considerable amount of reduction in the time complexity of solving K-SAT, whenever one can find a way to represent the formula in the way that imposes maximum restriction for variable assignment. Sometimes this type of forced assignment leading to the pruned search tree is called implication. The best example of this type of forced assignment can be observed in Horn theory [14].

It must be mentioned that in the case of unsatisfiable formula, reaching to conflict as soon as possible (in polynomial time) is realized as a sign of an effective search strategy. A conflict can be detected in the formula if at some point there is a clause in the formula with all of its literals evaluating to zero. The clause with this condition is called conflicting clause. A conflict in the formula happens as the result of earlier improper assignments.

Different strategies have been developed to escape from conflicts. Backtracking to the earlier assignments and change them in the controlled way, is the common them of all these strategies [11]. DPLL has experienced many significant improvements over the years based on these backtracking techniques. Conflict Driven Learning and Non-Chronological Backtracking are among the best improvements which have enhanced the power of DPLL algorithm in a considerable way [15], [16]. These improvements are based on the simple strategy of learning as much as possible from any conflict and its source in order to avoid it in the subsequent assignments.

Modern algorithms for solving SAT problem, gets the benefit of an improved type of unit clause rule, called two-literal watching and also improved technique in branching and variable assignment by considering the variables presented in recently conflicting clauses [17].Sometimes it is justifiable to apply random restart technique due to the complications associated with Conflict Driven Learning techniques and correlations among different clauses [18].

There is an interpretation of K-SAT which puts it in the category of discrete optimization problems. In this perspective, we are looking to maximize the number of satisfied clauses. This maximum can reach to the total number of clauses in which satisfaction happens. Therefore it is possible to use discrete optimization techniques like Simulated Annealing [19], Tabu Search [20], Neural Network [21] and Genetic Algorithms [22] to solve it.

Realization of K-SAT as an optimization problem refreshes our mind about the general difficulties of finding the maximum of the objective function. The intractable nature of the problem exhibits itself as the difficulties of bypassing the exponential number of local maximums or local minimums in its objective function [23]. Considering the K-SAT as an optimization problem, one can use stochastic local search algorithms to bypass the pathological difficulties of finding the global maximum of its objective function.

Stochastic local search algorithms were used for the first time by Minton et al. [24] for solving constraint satisfaction problem and for MAX-SAT problem by Hansen and Janmard [25]. Particularly for K-SAT, stochastic local search was used by Gu [26] and Selman et al. [27]. Selman et al. introduced GSAT algorithm which was more effective than DPLL variants used in those days and their approach sparked considerable interest in Artificial Intelligence Community.

In spite of all efforts, up to now, we don't have a polynomial time algorithm for solving K-SAT. On the other side there is a belief supported by many practical experiments which asserts that exponential time complexity occurs for a limited sub-space of K-SAT instances [28]. After the seminal work of Cheesman et al. [29], today we know that hard instances for K-SAT reside at the threshold of satisfiable to unsatisfiable phase, which occurs at specific value of $\frac{m}{n}$ known as α_s(here m is the number of clauses or constraints and n is the number of variables in the K-SAT formula). Theoretical investigations about the source and nature of this phase transition in the K-SAT problem have revolutionized our understanding from this problem and its state space geometry. Recently threshold conjecture has been proved analytically for some specific conditions [30].

The effectiveness of applying statistical physic methods for analyzing the source of phase transition in the K-SAT problem, has provided us a very detailed picture of solution space upon which many other thresholds of transition have been recognized. In addition to α_S(Satisfiable to Unsatisfiable threshold), an algorithmic threshold α_A is defined in such a way that all known algorithms running in polynomial time fail to find solution for $\alpha > \alpha_A$. Generally it is known that $\alpha_A < \alpha_S$ [31].

Therefore the previous satisfiable phase is partitioned into different regions in the light of new detailed picture of the solution space. Generally as α is increased the clusters of solutions in the solution space shrink and the connectivity among them is lost [32]. Let w and z be two distinct solutions in the set of all satisfying assignments for a specific K-SAT instance. A step [33] is defined as the number of variables which must be inverted in w to produce some w' that is also in a solution space. A path from w to z is defined as a sequence of steps starting with w and ending with$z$.

Intuitively we expect that increasing $\alpha = \frac{m}{n}$ decreases the number of solutions up to reaching to the unsatisfiability region but this phase transition is accompanied with other micro-transitions especially in the solution space of the problem.

For example bellow α_S for $\alpha \ll \alpha_S$, we observe connectivity in solution space. This connectivity exhibits itself as the existence of path between any two arbitrarily chosen solutions w and z.This connectivity is transformed at some specific value, called $\alpha_d < \alpha_S$ in which the solutions are partitioned into different clusters. Therefore at this value α_d, the condition can be described technically by the following equation[33].

$$S = \cup_{i=1}^{p} S_i \ such\ that\ \forall m, l \in \{1, \dots, p\}: S_m \cap S_l = \emptyset\ \&\ \nexists\ path\ from\ w\ to\ z\ if\ w \in S_m\ and\ z \in S_l\ and\ m \neq l \quad (4)$$

Further increment of α changes the size and the number of clusters into exponentially large number (in the problem size n) where two solutions belonging to different clusters, have a large hamming distance that scales with problem size [34].

There are several other phase transitions which can be defined for the topological transformation of the solutions in the state space [33], [35]. For example one can also identify α_c in which a condensation takes place such that for any $\alpha > \alpha_c$, the majority of solutions belong to sub-exponential number of clusters. Generally we have: $\alpha_A < \alpha_d < \alpha_S$ [36], [37].

In this paper by considering this picture of solution space, it is tried to improve randomized algorithm presented by Schoning [38] in 1999 and reduce its time complexity.

III. Randomized Algorithm and Its Analysis

When the structure of K-CNF formula cannot provide an insight for pruning the solution space, nothing can do better than random search [23]. Theoretically the maximum amount of information can be extracted by this strategy from state space of the problem. The first successful randomized local search was introduced by Schoning [38]. His method has been based on random walking in the space of possible assignments starting from random truth assignment and flipping the suitable literals until the formula gets satisfied.

Taking into account different thresholds mentioned in the previous section, we expect to find a solution in polynomial time when $\alpha < \alpha_A$. The situation would be harder for α near α_d and after it due to the clusterization. Because of the interaction between different clauses, up to now, no one has given an analytic model which makes us able to count the number of unsatisfied clauses during the process of random walking in the solution space. Therefore the performance of Schoning's algorithm is analyzed by focusing on the hamming distance between current assignments namely B and one particular satisfying assignment called A^*. Let's look at the Schonin's algorithm [38].

This algorithm starts at a uniformly random truth assignment B. If B satisfies the formula, it would be returned. Otherwise the algorithm repeatedly chooses a clause c from unsatisfied clauses, then chooses a variable x uniformly from c's literals and flipping it until the formula is satisfied or the algorithm runs out of time.

As a matter of fact, starting from random initial assignment B , the algorithm tries to reduce the hamming distance between current assignment and satisfying assignment by random flipping of variables in unsatisfied clauses. Obviously $Hamming - distance(B, A^*)$ would be less than n(the number of variables in the formula) and greater or equal to zero.

$$0 \leq Hamming - diatance(B, A^*) \leq n \quad (5)$$

Reaching to the zero hamming distance means the satisfying assignment has been found. Let's define $P(d)$ as the probability of reaching to the zero. We know that:

$$P(0) = 1\ \&\ \lim_{d \to \infty} P(d) = 0 \quad (6)$$

As a matter of fact we are looking for the evolution of $P(d)$ during the execution of randomized algorithm. Considering the algorithm, it is not hard to realize that flipping the variable chosen uniformly from the unsatisfied clause is responsible for the evolution of $P(d)$. Let c be the chosen clause from unsatisfied clauses with assumption that the formula is satisfied finally at threshold of α_S, one can expect that A^* agrees with c on at least one of its variables. Therefore flipping one variable would lead to reducing the hamming distance and moving toward A^*(satisfying assignment) with probability $\frac{1}{K}$. Consequently the hamming distance is increased due to this flipping with probability $(1 - \frac{1}{K})$. Now we have enough information to write the governing equation of $P(d)$ at the vicinity of α_S.

$$P(d) = \frac{1}{K} P(d-1) + \frac{K-1}{K} P(d+1) \quad (7)$$

Equation (7) reflects the evolution of $P(d)$ when we are dealing with highly constrained problem near α_S. For more relaxed type of problem in which α is around α_A , the agreement of A^* with c equals to l, where $l > 1$, due to the connectivity in the solution space among the satisfying assignments. Therefore, (7) can be transformed for covering more relaxed problems into (8).

$$P(d) = \frac{l}{K} P(d-1) + \frac{K-l}{K} P(d+1), k - l > 1\ and\ l > 1 \quad (8)$$

The boundary conditions $(P(0) = 1\ \&\ \lim_{d \to \infty} P(d) = 0)$ are still valid for relaxed type of problem. Solving (8) will give us the answer:

$$P(d, l) = \left(\frac{K-l}{l}\right)^{-d} \quad (9)$$

Equation (9) shows that the success of finding the satisfying assignment is completely controlled by the hamming distance of randomly chosen initial assignment with the satisfying assignment A^* and also by the probability of having suitable flipping of variables. In order to calculate the $P_{Success}$ (the probability of finding the satisfying assignment by algorithm), it is enough to divide the 2^n possible assignments into partitions with the same hamming distance to the desired satisfying assignment and compute the average of $P(d)$ over d.

$$Prob(Hamming - distance(B, A^*) = d) = 2^{-n} \binom{n}{d} \quad (10)$$

Therefore for computing $P_{Success}$ by applying the generalized type of Schoning's algorithm in which the random

walker reduces the hamming distance to the desired satisfying assignment with probability $\frac{l}{K}$ and increases it with probability$\frac{K-l}{K}$, we have the following equation:

$$P_{Success}(l) = \sum_{d=0}^{n} 2^{-n}\binom{n}{d}P(d,l) = \sum_{d=0}^{n} 2^{-n}\binom{n}{d}(\frac{l}{K-l})^{d}\,1^{n-d} = 2^{-n}(\frac{l}{K-l}+1)^{n} = 2^{-n}(\frac{K}{K-l})^{n} = (\frac{2(K-l)}{K})^{-n} \quad (11)$$

By applying amplification technique in order to get rid of emergent error associated with randomized algorithm [39], the time complexity of applying this algorithm for solving K-SAT problem is$T(n) = poly(n)(\frac{2(K-l)}{K})^{n}$. For computing this time complexity, it has been assumed that Schoning's algorithm needs polynomial time complexity.

Therefore boosting technique applied for reducing the error of randomized algorithm is the source of emerging exponential time complexity of the resulted algorithm. In the next section we argue how the parameter$l$, taking part in the probability of reducing hamming distance, can change our usual expectation from the complexity of this method.

IV. Conclusion and Future Work

In this paper we analyzed the consequences of applying Schoning's randomized search method [38] for values of$\alpha \ll \alpha_S$. Usually the performance of algorithms is analyzed in worst case, in which the maximum time complexity can be observed.

For K-SAT, as it has been shown by Cheeseman et al. [29], the worst instances happen at the onset of α_S in which one can expect that each clause is satisfied by a proper value of just one of its variables. In this highly constrained region, we have $l = 1$ and the time complexity of algorithm is $(\frac{2(K-1)}{K})^{n}Poly(n)$. By applying an information theoretic method [40] to calculate the entropy of the random walk in this case (where$P(\Delta d = +1) = \frac{K-1}{K}$ & $P(\Delta d = -1) = \frac{1}{K}$) we reach to (12).

$$H[P_{Worst}] = \frac{1}{K}\log_2 K + \frac{K-1}{K}\log_2 \frac{K}{K-1} \quad (12)$$

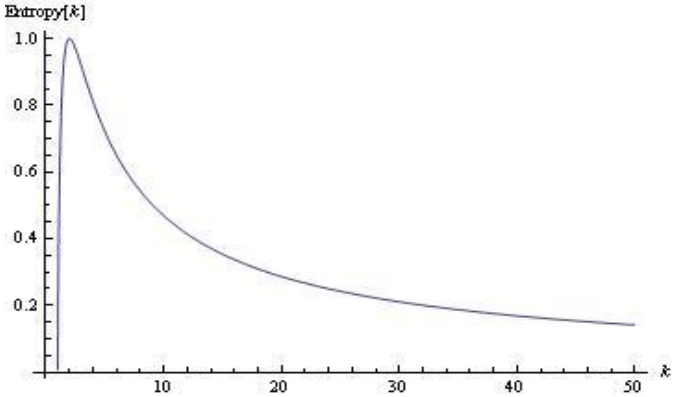

Fig. 1. The Entropy of random walking in the state space at the vicinity of α_S.

Fig. 1 shows the entropy function of K-SAT problem at the threshold of α_S in which satisfaction comes in highly constrained manner. Remember that for $\alpha > \alpha_S$ the K-SAT would be unsatisfiable.

A deeper look at Fig. 1 shows that maximum value of entropy function for k-SAT occurs at k=2. We know that k-SAT is solvable in polynomial time for k=2. It means that, the exponential size of solution space is pruned maximally when the amount of information gained from random walk in solution space becomes maximum at k=2.

In Fig. 2, the entropy function of random walking in the solution space of more relaxed situation in which $P(\Delta d = +1) = \frac{K-l}{K}$ & $P(\Delta d = -1) = \frac{l}{K}$ where $l > 1$, has been depicted. As it can be observed, for larger value of$l$, which is seen in more relaxed problem ($\alpha \approx \alpha_A$), the maximum value of function is shifted toward larger values of$K$.

It has been known for several years that random walking in the solution space is the best strategy in the lack of any guide for pruning the large state spaces [23]. The result of this paper approves this hypothesis. When the entropy of random walk is maximized the problem can be solved effectively in polynomial time due to the vanishing of exponential part of time complexity. Obviously $T(n) = poly(n)(\frac{2(K-l)}{K})^{n}$ is transformed to $poly(n)$ at $l = K/2$, in which maximum entropy of random walk is happened and deviation from this maximum entropy of random walk would be accompanied by the emergence of exponential time complexity.

Although l is known to be larger than 1 for$\alpha \ll \alpha_S$, It is an open question to find a strict mathematical bound for it. This trend of study will improve our understanding in the future.

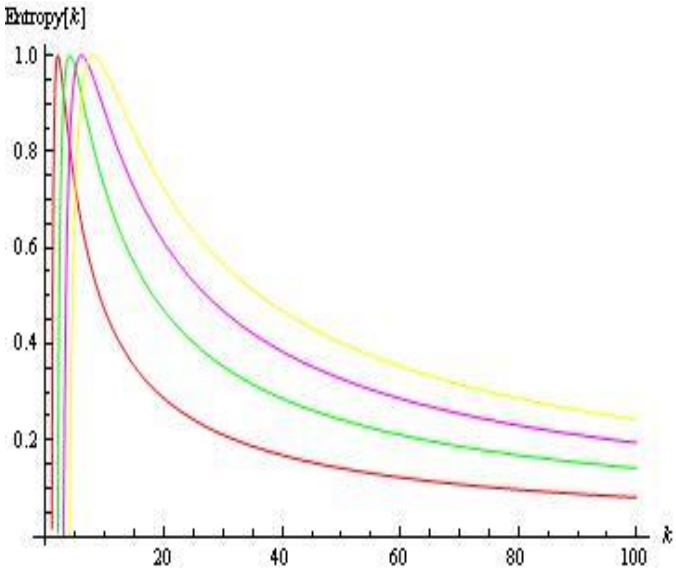

Fig. 2. The entropy of random walk for different values of l.$l = 1$with red line,$l = 2$ with green line,$l = 3$ with magenta line and $l = 4$ with yellow line.

REFERENCES

[1] H. Zhang, "Generating college conference basketball schedules by a SAT solver," In Proceedings of the 5th International Symposium on Theory and Applications of Satisfiability Testing, Cincinnati, USA, 2002.

[2] H. Kautz, B. Selman, "Planning as satisfiability," European Conference on Artificial Intelligence, Paris, France, 1992.

[3] D. Jackson, M. Vaziri, "Finding bugs with a constarint solver," In Proceeding of The International Symposium on Software Testing and Analysis, Portland, USA, 2000.

[4] G-j. Nam, K. A. Sakallah, R. A. Rutenbar, "Satisfiability-based layout revisited: detailed of complex FPGAs via search-based Boolean SAT," International Symposium on Field-programmable Gate Arrays, California, USA, 1999.

[5] S. Narain, G. Levin, V. Kaul, S. Malik, "Declarative infrastructure configuration and debugging," Journal of Network Systems and Management, Special issues on Security Configuration, 2008.

[6] A. Biere, A. Cimatti, E. M. Clarke, Y. Zhu, " Symbolic model checking without BDDs. tools and algorithms for the analysis and construction of systems," Proceedings of the 5th International Conference on Tools and Algorithms for Construction and Analysis of Systems, pp. 193-207, Netherland, 1999.

[7] E. M. Clarke, O. Grumberg, D. A. Peledi, "Model Checking," Cambridge, Massachussetts, MIT Press, 1999.

[8] K. L. Mcmilan, "Applying SAT methods in unbounded symbolic model checking," 14th International Conference on Computer Aided Verification, London, UK, 2002.

[9] S. A. Cook, "The complexity of theorem proving procedures," Third Annulal ACM Symposium on Theory of Computing, USA, 1971.

[10] M. R. Garey, D. S. Johnson, "Computers and Interactability: A Guide to the Theory of NP-Completeness," W. H. Freeman, 1979.

[11] R. Marino, G. Parisi, "The backtracking survey propagation algorithm for solving random K-SAT problems," Nature Communication, vol. 7, pp. 12996, 2016.

[12] M. Davis, G. Logemann, D. Loveland, "A machine program for theorem proving," Communication of ACM, vol. 5, pp. 394-397, 1962.

[13] M. Davis, H. Putnam, "A computing procedure for quantification theory," Journal of ACM, vol. 7, pp. 201-215, 1960.

[14] V. W. Marek, "Introduction to Mathematics of Satisfiability," CRC Press, 2009.

[15] R. Bayardo, R. Schrag, "Using CSP look-back techniques to solve real-world SAT Istances," National Conference on Artificial Intelligence, USA, 1997.

[16] J. P. Marques-Silva, K. A. Sakallah, "Conflict analysis in search algorithms for propositional satisfiability," IEEE International Conference on Tools with Artificial Intelligence, USA, 1996.

[17] M. W. Madigan, C. F. Madigan, Y. Zhao, L. Zhang, S. Malik, "Chaff: engineering an efficient SAT solver," 38th Conference on Design Automation, New York, USA, 2001.

[18] C. P. Gomes, B. Selman, H. Kautz, "Boosting combinatorial search through randomization," National Conference on Artificial Intelligence, Madison, USA, 1998.

[19] W. M. Spears, "Simulated annealing for hard satisfiability problems: Cliques, Coloring and Satisfiability," In: Johnson, D. S. and Trick, M. A. (Eds.), Second DIMACS Implementation Challenge, American Mathematical Society, pp. 533-558, 1993.

[20] B. Mazure, L. Sas, E. Grgoire, "Tabu search for SAT," 14th National Conference on Artificial Intelligence, Providence, USA, 1997.

[21] W. M. Spears, "A Neural Network algorithm for Boolean satisfiability problems," International Conference on Neural Network, USA, 1996.

[22] E. Marchiori, C. Rossi, "A flipping Genetic Algorithm for hard 3-SAT problems," Genetic and Evolutionary Computation Conference, Orlando, USA, 1999.

[23] H. H. Holger, S. Thomas, "Stochastic Local Search: Foundations and Applications," Elsevier, 2005.

[24] S. Minton, M. D. Johonston, A. B. Philips, P. Laired, "Solving large-scale constraint satisfaction and scheduling problems using a Heuristic Repair Method," 8th National Conference on Artificial Intelligence, Menlo Park, California, USA, 1990.

[25] B. Hansen, B. Janmard, "Algorithms for the Maximum Satisfiability Problem," Journal of Computing, vol. 44, pp. 279-303, 1990.

[26] J. Gu, "Efficient local search for very large scale satisfiability problems," SIGART Bulletin, vol. 3, no. 1, pp. 8-12, 1992.

[27] B. Selman, H. Levesque, D. G. Mitchell, "A new method for solving hard satisfiability problems," 10th national Conference on Artificial Intelligence, Menlo Park, California, USA, 1992.

[28] B. Selman, D. G. Mitchell, H. Levesque, "Generating hard satisfiability problems," Artificial Intelligence, vol. 81, pp. 17-29, 1996.

[29] P. Cheeseman, B. Kanefsky, W. Taylor, "Where the really hard Problems are," In Proceedings of the 11th IJCAI, USA, 1991.

[30] J. Ding, A. Sly, N. Sun, "Proof of the satisfiability conjecture for large K," 47th Annual ACM Symposium on Theory of Computing, Portland, USA, 2015.

[31] A. Montanari, F. Ricci-Tersenghi, G. Semerjian, "Clusters of solutions and replica symmetry breaking in random K-Satisfiability," Journal of Statistical Mechanics, pp. 04004, 2008.

[32] M. Mezard, A. Montanari, "Information, Physics and Computation," London, Cambridge University Press, 2009.

[33] M. Mezard, T. Mora, R. Zecchina, "Clustering of solutions in the random satisfiability problem," Physical review Letters, vol. 94, no. 19, pp. 0200, 2005.

[34] M. Mezard, G. Parisi, R. Zecchina, "Analytic and algorithmic solution of random satisfiability problems," Science, vol. 297, pp. 812-815, 2002.

[35] L. Zdeborova, F. Krzakala, "Phase Transitions in the coloring of random graphs," Physical Review E, vol. 76, pp. 031131, 2007.

[36] D. Achlioptas, A. Coja-Oghlan, "Algorithmic barriers from Phase Transitions," 49th IEEE Symposium on Foundations of Computer Science, Philadelphia, USA, 2008.

[37] D. Achlioptas, A. Coja-Oghlan, F. Ricci-Tersenghi, "On the solution-space geometry of random Constraint Satisfaction Problems," Random Structure and Algorithms, vol. 38, pp. 251-268, 2011.

[38] U. Schoning, "A probabilistic algorithm for K-SAT and Constraint Satisfaction Problems," 40th IEEE Sumposium on Foundations of Computer Science, USA, 1999.

[39] J. Hromkovic, "Design and Analysis of Randomized Algorithms: Introduction to Design Paradigms," Berlin, Heidelberg, New York: Springer, 2005.

[40] T. M. Cover, J. A. Thomas, "Elements of Information Theory," Second edition, New Jersey, John Wiley & Sons, 2006.