REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION		1b. RESTRICTIVE MARKINGS			
UN					
2a.		3. DISTRIBUTION/AVAILABILITY OF REPORT			
2b. AD-A168 617		Approved for public release; distribution unlimited.			
4.		5. MONITORING ORGANIZATION REPORT NUMBER(S)			
		AFOSR-TR- 86-0310			
6a. NAME OF PERFORMING ORGANIZATION Indiana University	6b. OFFICE SYMBOL (If applicable)	7a. NAME OF MONITORING ORGANIZATION Air Force Office of Scientific Research			
6c. ADDRESS (City, State and ZIP Code) Department of Computer Science Indiana University Bloomington, Indiana 47405		7b. ADDRESS (City, State and ZIP Code) Directorate of Mathematical & Information Sciences, Bolling AFB DC 20332-6448			
8a. NAME OF FUNDING/SPONSORING ORGANIZATION AFOSR	8b. OFFICE SYMBOL (If applicable) NM	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER AFOSR-84-0372			
8c. ADDRESS (City, State and ZIP Code)		10. SOURCE OF FUNDING NOS.			

		PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT NO.
Bolling AFB DC 20332-6448		61102F	2304	A2	

11. TITLE (Include Security Classification)
PROBABILISTIC ANALYSIS OF ALGORITHMS FOR NP-COMPLETE PROBLEMS

12. PERSONAL AUTHOR(S)
Professor John Franco

13a. TYPE OF REPORT Final	13b. TIME COVERED FROM 9/30/84 TO 9/29/85	14. DATE OF REPORT (Yr., Mo., Day) 1985, December 12	15. PAGE COUNT 6

16. SUPPLEMENTARY NOTATION

17.	COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB. GR.	

19. ABSTRACT (Continue on reverse if necessary and identify by block number)

The probabilistic performance of a number of algorithms for the Satisfiability Problem (SAT) has been investigated analytically and experimentally using a constant-clause-size model generating n clauses of k literals taken from r variables as well as a constant-density model generating n clauses containing each of r variables independently with probability p. In the case of the constant-density model one algorithm has been shown to solve SAT in polynomial time with probability approaching 1 as n and r get large when $p > \ln(n)/r$ and another has been shown to solve SAT in polynomial time with probability approaching 1 as n and r get large when $p < \ln(n)/(2r)$. In the case of the constant-clause-size model the unit clause heuristic has been shown to be effective, in probability,

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ☒ SAME AS RPT. ☐ DTIC USERS ☐	21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Captain John P. Thomas	22b. TELEPHONE NUMBER (Include Area Code) (202) 767-5026	22c. OFFICE SYMBOL NM

86 6 10 121

UNITED STATES AIR FORCE
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH
BUILDING 410, BOLLING AFB, D.C. 20332
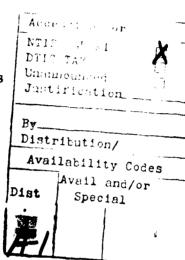
Grant No. AFOSR 84-0372

ANNUAL SCIENTIFIC REPORT

Probabilistic Analysis of Algorithms for NP-complete Problems

John Franco, Principal Investigator
Department of Computer Science
Indiana University
Bloomington, Indiana 47405

Abstract

The probabilistic performance of a number of algorithms for the Satisfiability Problem (SAT) has been investigated analytically and experimentally using a constant-clause-size model generating n clauses of k literals taken from r variables as well as a constant-density model generating n clauses containing each of r variables independently with probability p. In the case of the constant-density model one algorithm has been shown to solve SAT in polynomial time with probability approaching 1 as n and r get large when $p > \ln(r)/r$ and another has been shown to solve SAT in polynomial time with probability approaching 1 as n and r get large when $p < \ln(n)/(2r)$. In the case of the constant-clause-size model the unit clause heuristic has been shown to be effective, in probability, when $\lim_{n,r\to\infty} n/r < 2^{k-1}((k-1)/(k-2))^{k-2}/k$ and a generalization of the unit clause heuristic has been shown to be effective, in probability, when $\lim_{n,r\to\infty} n/r < 3.09 \cdot 2^{k-2}((k-1)/(k-2))^{k-2}/(k+1)$ for $3 < k < 40$. When $k = 3$ the unit clause heuristic with the next variable given an assignment which satisfies the maximum number of clauses has been shown effective, in probability, when $\lim_{n,r\to\infty} n/r < 3$. The analysis of these heuristics involves solving sets of differential equations which model the "flow" of clauses from sets of clauses containing many literals to sets containing few literals. Similar differential equations have been developed for the Chromatic Number problem and Set Covering, both of which are NP-complete. Also, the probabilistic performance of two algorithms for the Maximum 2-Satisfiability problem has been obtained.

1. Research Objective

The goal of this research is to develop and analyze algorithms which can, in some practical sense, solve certain NP-complete problems efficiently. By solve we mean determine whether a solution to a given instance of an NP-complete problem exists where, for the problems we have considered, a solution is an assignment of values to a list of variables which cause some predicate to be true. We do not consider actually finding solutions when they exist since doing so adds unnecessary complexity to the statement of the algorithms: the algorithms we consider can all be modified to find solutions without significantly altering performance. NP-complete problems are found in Crytology, Operations Research, Artificial Intelligence, Computer System Design and many other areas. There is no known algorithm for any NP-complete problem which runs in time bounded by a polynomial on the length of the input (polynomial time) in the worst case nor is one likely to be found. We seek algorithms which solve nearly every instance of specific NP-complete problems in polynomial time.

To prove an algorithm A solves nearly every instance of a specific problem X in polynomial time we establish a probability distribution $D(n)$ on instances of X of "size" n (referred to as a model) and then show that A solves a random instance of X generated according to $D(n)$ in polynomial time with probability approaching 1 as n approaches infinity; then A is said to solve X efficiently in probability. Usually the proof holds only under certain conditions. Sometimes, when $D(n)$ is such as to heavily favor the generation of instances with solutions, the weaker result that A "proves" the existence of a solution to a random instance of X in polynomial time with probability bounded from below by a constant greater than zero is obtained instead; then A is said to solve X efficiently with bounded probability. Again, the result holds only under certain conditions (one condition that must be satisfied is that nearly all random instances generated according to $D(n)$ have a solution). The algorithms that we consider here "prove" the existence of a solution by repeatedly choosing a variable and an assignment to that variable until the predicate is true: at each step the possible choices are ranked based on some heuristic and the top ranked possibility is chosen. If the predicate cannot be made true the algorithm stops without solving the instance. For the kinds of algorithms and distributions we consider, if A solves X efficiently with bounded probability under some set of conditions then we may regard this as strong evidence that the *Backtrack* algorithm, using the heuristics of A to decide the order in which to consider variables and assign values, solves X efficiently in probability under the same set of conditions.

1

The NP-complete problem we are primarily interested in is the Satisfiability problem (SAT). An instance I of SAT is a boolean expression in conjunctive normal form and a solution to that instance, if one exists, is a truth assignment to the variables in I which cause I to have value true; such a truth assignment is said to satisfy I. SAT remains NP-complete even if all disjunctions contain as few as three literals. SAT is closely related to problems in Artificial Intelligence as well as other NP-complete problems. Algorithms which solve SAT efficiently in some probabilistic sense will, with slight modification, probably solve other NP-complete problems efficiently in the same probabilistic sense.

2. Status of the Research

Although there has been a significant level of research activity in this area no one has succeeded in getting the results we have obtained for algorithms designed to solve instances of SAT efficiently in some probabilistic sense.

Two models have been used for analysis: one is a constant-clause-size model and the other is a constant-density model. According to the constant-clause-size model a random instance of SAT contains n clauses (disjunctions) selected independently and uniformly from the set of all possible disjunctions containing exactly k literals which can be composed from r boolean variables under the restriction that no two literals in the same disjunction are associated with the same variable. We are interested in the case $k \geq 3$ since SAT is NP-complete if clauses are allowed to have three or more literals. For the special case $k = 3$ SAT is called 3-SAT. According to the constant-density model a random instance of SAT contains n clauses each generated independently as follows: for each of r variables (a) place into the clause, with probability $p/2$, the uncomplemented literal associated with the variable, (b) place into the clause, with probability $p/2$, the complemented literal associated with the variable and (c) place neither complemented nor uncomplemented literal into the clause with probability $1 - p$.

We have shown that two algorithms solve SAT efficiently in probability under the constant-density model when n and r are polynomially related. More specifically, one algorithm finds a solution to a random instance of SAT in probability when $p > \ln(n)/r$ and the other verifies that a random instance of SAT has no solution in probability when $p < \ln(n)/(2r)$. Thus, under the constant-density model, SAT is solved efficiently in probability for all but a vanishingly small range of values of p if n and r are polynomially related (it is easy to see why this is a reasonable restriction). These results were written up in "On the Probabilistic Performance of Algorithms for the Satisfiability Problem" which has been accepted for publication in *Information Processing Letters*.

2

Although these results are theoretically interesting they have little practical meaning since the two algorithms are trivial and are not likely to be effective in practice. In fact, the results suggest that the model used is faulty since it generates a large number of random instances of SAT which can be trivially solved (only local information is necessary to solve the vast majority of these instances). The issue of choosing a "reasonable" model was discussed in "Sensitivity of Probabilistic Results on Algorithms for NP-Complete Problems to Input Distributions" which appearred in *SIGACT NEWS* **17**,1 (1985) pp. 40-59.

The results obtained for the constant-clause-size model are probably more meaningful. Under this model it was shown that the unit clause and maximum occurring literal selection heuristics are effective, with bounded probability, in finding solutions to random instances of 3-SAT. According to these heuristics, the next variable to be given a value is taken from a unit clause, if one exists, and is assigned a value which satisfies that clause; otherwise a variable v is chosen randomly from the set of unassigned variables and is assigned the value *true* if literal v appears in more remaining clauses than literal $\bar{v}$, is assigned the value *false* if literal $\bar{v}$ appears in more remaining clauses than literal v and is assigned value *true* with probability 1/2 or value *false* with probability 1/2 if the number of clauses containing v and $\bar{v}$ is the same. We have shown that these heuristics find solutions to random instances of 3-SAT efficiently in probability when $n/r < 3$. This is interesting since nearly all random instances generated when $n/r > 4$ are unsatisfiable.

Also interesting is the analysis of these heuristics. As variables are assigned values some clauses are satisfied and some literals are falsified. The satisfied clauses are never considered again and the clauses containing the falsified literals may be regarded as clauses containing one fewer literal. For all $0 \leq i \leq 3$ let $C_i(j)$ be the subset of unsatisfied clauses containing i unassigned literals prior to selecting the jth variable to be assigned a value. As the algorithm proceeds, clauses flow from $C_i(j)$ to $C_{i-1}(j+1)$ or out of the system for $i = 1, 2, 3$. This flow is modeled by a system of differential equations which are solved giving the flow into $C_1(j)$ for all $1 \leq j \leq r$. If this flow remains less than 1 it is not likely that the algorithm will fail to produce a solution since the number of unit clauses at any time is then unlikely to be higher than a constant so the probability that a pair of complementary unit clauses exists (the condition which would prevent any extention of the current partial truth assignment from satisfying the given instance) is low. Conditions on n and r which insure that the flow into $C_1(j)$ is less than 1 for all j have been found. This analysis as well as the results represent significant work that is written up in "Probabilistic Analysis of Two Heuristics for the 3-Satisfiability Problem" which has been accepted for publication in *SIAM Journal on Computing*.

Using the idea of flow analysis, a generalization of the unit clause heuristic was analysed under the constant-clause-size model. According to this heuristic the next variable to be assigned a value is chosen from a clause containing the smallest number of unassigned literals and it is assigned the value which satisfies the clause from which it was chosen. It was found that for $3 < k < 40$ this heuristic is efficient in probability when $n/r < 1.845 \cdot 2^{k-2}((k-1)/(k-2))^{k-2}/(k+1)$ and is efficient with bounded probability when $n/r < 3.09 \cdot 2^{k-2}((k-1)/(k-2))^{k-2}/(k+1)$. These results have been written up in "Probabilistic Analysis of a Generalization of the Unit Clause Literal Selection Heuristic for the k-Satisfiability Problem" which was presented at the *Symposium on Approximately Solved Problems*, Columbia University, New York (1985). This paper has also been submitted to the *Journal of the Association for Computing Machinery*.

We have also applied flow analysis to algorithms for the NP-complete Chromatic Number and Set Covering problems. Unfortunately the resulting systems of equations are nonlinear so we have not yet obtained good solutions to them.

We have also applied flow analysis to algorithms for the Maximum 2-SAT problem. An instance of Maximum 2-SAT is a boolean expression in conjunctive normal form such that each clause contains 2 literals. The problem is to find the maximum number of clauses that can be satisfied by any truth assignment to the variables of the instance. This problem is NP-complete. The best known approximation algorithm for this problem is guaranteed to find at least 75 percent of the clauses satisfied. However we have shown that two algorithms find a much larger fraction of simultaneously satisfiable clauses than this in probability.

3. Interpretation of Results

The constant-clause-size model seems to generate non-trivial instances of SAT since simple-minded algorithms which work so well on instances generated by the constant-density model do not work at all well on random instances generated according to the constant-clause-size model when the limiting ratio of n/r is fixed (i.e. a function of k). The case of the limiting ratio of n/r being fixed is important since random instances are "hardest" when the probability that a solution exists is about $1/2$ and this occurs when the limiting ratio is fixed. Despite the relatively "hard" instances generated by the constant-clause-size model a number of algorithms have been shown to "prove" that a solution to a given random instance I of SAT exists for nearly every I that has a solution when $k = 3$; these algorithms are not quite as effective for arbitrary k.

Perhaps surprising is the difference in the range of n/r over which algorithms perform well probabilistically. In particular, the unit clause heuristic and the generalized unit clause heuristic are not much different in structure but the bound on the limiting ratios of n/r for which good probabilistic performance is achieved is larger for the generalized unit clause heuristic by a factor of 2. Furthermore, from a previous result, the bound on ratios n/r for which good probabilistic performance of the pure literal heuristic is achieved does not even increase with k while the bounds for the heuristics studied here are all exponential in k.

We have been able to rank a number of algorithms for solving SAT by their probabilistic performance. One of these algorithms has been shown experimentally to be extemely effective on instances of 3-SAT when those instances have solutions. We have not yet succeeded in producing an algorithm for SAT which, under the constant-clause-size model, is effective in determining that no solution exists when its input is an instance with no solution. This is the next step in this research.

5

4. Publications under Grant No. AFOSR 84-0372

[1] J. Franco, "Sensitivity of probabilistic results for algorithms for NP-complete problems to input distribution," *SIGACT News* **17**,1 (1985) pp. 40-59.

[2] M.T. Chao and J. Franco, "Probabilistic analysis of two heuristics for the 3-Satisfiability problem," to appear in *SIAM J. Comput.*

[3] M.T. Chao and J. Franco, "Probabilistic analysis of a generalization of the unit clause literal selection heuristic for the k-Satisfiability problem," submitted to *J. ACM*.

[4] J. Franco, "On the probabilistic performance of algorithms for the Satisfiability problem," to appear in *Info. Proc. Letters*.

[5] E. Choukmane and J. Franco, "An approximation algorithm for the Maximum Independent Set problem," to appear in *Networks*.

END

DTIC

7 - 86