Easy Problems are Sometimes Hard

Ian P. Gent
Department of Artificial Intelligence
University of Edinburgh
80 South Bridge
Edinburgh EH1 1HN, United Kingdom.
I.P.Gent@edinburgh.ac.uk

Toby Walsh
INRIA-Lorraine
615, rue du Jardin Botanique
54602 Villers-les-Nancy, France.
walsh@loria.fr

June 27, 1993

Abstract

We present a detailed experimental investigation of the phase transition from satisfiable to unsatisfiable for randomly generated instances of SAT problems. This study demonstrates that problems in classes previously thought to be 'easy' can sometimes be orders of magnitude more difficult than the worst problems in classes considered 'hard'. This result explains the large variability in problem difficulty observed in these 'easy' classes. We present evidence which suggests that this phenomenon cannot be eliminated by the use of better heuristics. These difficult problems appear to be fundamentally hard or to contain subproblems that are fundamentally hard.

In practice such a phenomenon would be attributed to an "experimental error" or be discarded as an "outlier". It is difficult to see what one does not expect to see.

William Feller, *An Introduction to Probability Theory and Its Applications*, Third Edition.

1 Introduction

One of the most difficult problems facing researchers interested in algorithms for NP-complete problems is how to compare them. Theoretical results are often difficult to obtain and may only apply to problem classes chosen more for their amenability to analysis than for their difficulty. Empirical results, on the other hand, require a test set which may be just as unrepresentative of typical problems. Thus in constructing an empirical *science* of algorithms, it is important to find good sources of test problems with known characteristics. This has led to recent investigations into techniques for randomly generating hard problems. In this paper, we contribute to these investigations by showing that as well as regions in which problems are usually hard, random problem classes can contain regions in which problems are usually *easy* but sometimes extraordinarily *hard*.

The major empirical observation we follow was made by Cheeseman et al [1]. Many techniques for generating problems randomly depend on a natural parameter; for example the average connectivity of a graph of fixed size, or the number of clauses in a clause set containing a fixed number of variables. As this parameter is varied, there tends to be a phase transition from problems which are almost always satisfiable to a region where they are almost always unsatisfiable. Furthermore this phase transition is typically associated with problems that are very much more difficult to solve than problems away from the transition.

For SAT, the problem of determining the satisfiability of propositional formulae, considerable research has identified very precisely that random 3-SAT (this and other problem classes are described in §2) undergo a phase transition when there about 4.25 clauses for every variable [13, 12, 3]. This is generally considered a good method of generating hard instances of SAT. By comparison, the constant probability model is considered to give easy SAT problems since various theoretical results have shown that certain cases can be solved in polynomial average time [7, 11]. In addition, limited experimental studies have reached similar conclusions [13].

One of the aims of this paper is to demonstrate that these conclusions are premature. We found one 100 variable problem generated using the constant probability model that required over 350 million branches to solve using a simplified Davis-Putnam procedure. This was considerably more than the total number of branches required by the same procedure on a *thousand* 100 variable problems from the hard region of random 3-SAT. We observed that, whilst typical problems generated by the constant probability model are indeed easy, there is enormous variability between problems generated in the same way. Furthermore, the greatest variability occurs not in a region where about 50% of problems are satisfiable, but in a region where nearly 100% are, and it occurs with both satisfiable and unsatisfiable problems. As predicted by Cheeseman et al, though, it does happen at the phase transition from satisfiable to unsatisfiable problems.

To investigate the constant probability model in detail, we have performed a large number of experiments using a variety of different parameters for problem generation. Our results are as follows. In the constant probability model, problems of enormous difficulty for backtracking algorithms arise in a region where problems generated have at least a 90% chance of being satisfiable. Such hard problems are comparatively rare, but seem to increase in frequency with problem size, and increase in difficulty very quickly as problems become bigger. We have observed this behaviour in three different variants of algorithms: a simplified version of the Davis-Putnam procedure with backtracking and unit propagation only, the full Davis-Putnam procedure, and Davis-Putnam using the Jeroslow-Wang heuristic for selection of a literal in the splitting rule. Furthermore, we have observed that the same problem can vary enormously in difficulty depending merely on its presentation. Finally, we have also observed that at a region of approximately 50% chance of satisfiability, the average problem difficulty reaches a local maximum but with very much smaller variability, and that the median (but not the mean) problem difficulty is larger than in the 90% region.

2 Propositional Satisfiability, SAT

SAT, the problem of deciding if there is an assignment for the variables in a propositional formula that makes the formula true, is an interesting problem since it was the first computational task shown to be NP-hard [2]. It is also useful since many AI problems can be encoded quite naturally in SAT (*eg.* constraint satisfaction, diagnosis and vision interpretation [14]). In addition, satisfiability is closely related to theorem proving; refutation procedures call upon the fact that φ is derivable from Σ iff $\Sigma \cup \neg\varphi$ is not satisfiable.

In this paper we consider SAT problems as conjunctive normal form (CNF) formulae; a formula, Σ is in CNF iff it is a conjunction of clauses, where a clause is a disjunction of literals. In particular, we consider CNF formulae randomly generated according to two different models: random k-SAT and the constant probability model with empty and unit clauses omitted.

Problems in random k-SAT with N variables and L clauses are generated as follows: a random subset of size k of the N variables is selected for each clause, and each variable is made positive or negative with probability $\frac{1}{2}$.

In the constant probability model, clause sets with N variables and L clauses are generated according to a parameter p, $0 < p \leq 1$. For each clause, each literal (that is, a variable or the negation of a variable) is included with probability p, independently of the inclusion of other literals. In particular, for example, the inclusion of tautologous clauses (containing both a literal and its negation) is allowed. In this paper we use a variant of the constant probability model proposed in [9] in which if a clause is generated containing either *no* literals or only *one* literal, it is discarded and another clause generated in its place. We

procedure DP(Σ)
  if Σ empty **then return** satisfiable
  if Σ contains an empty clause **then return** unsatisfiable
  (Tautology rule)
  if Σ contains a tautologous clause c **then**
    return DP($\Sigma - \{c\}$)
  (Unit Propagation)
  if Σ contains a unit clause c **then**
    return DP(Σ simplified by assigning truth value which satisfies c)
  (Pure-literal Deletion)
  if Σ contains a literal l but not the negation of l **then**
    return DP(Σ simplified by assigning truth value which satisfies l)
  (Split-rule)
  if DP(Σ simplified by assigning a variable arbitrarily) is satisfiable **then**
    return satisfiable
    else return DP(Σ simplified by assigning variable opposite value)

Figure 1: The Davis-Putnam Procedure

do this because the inclusion of empty or unit clauses typically makes problems easier. We shall call this the "CP" model.

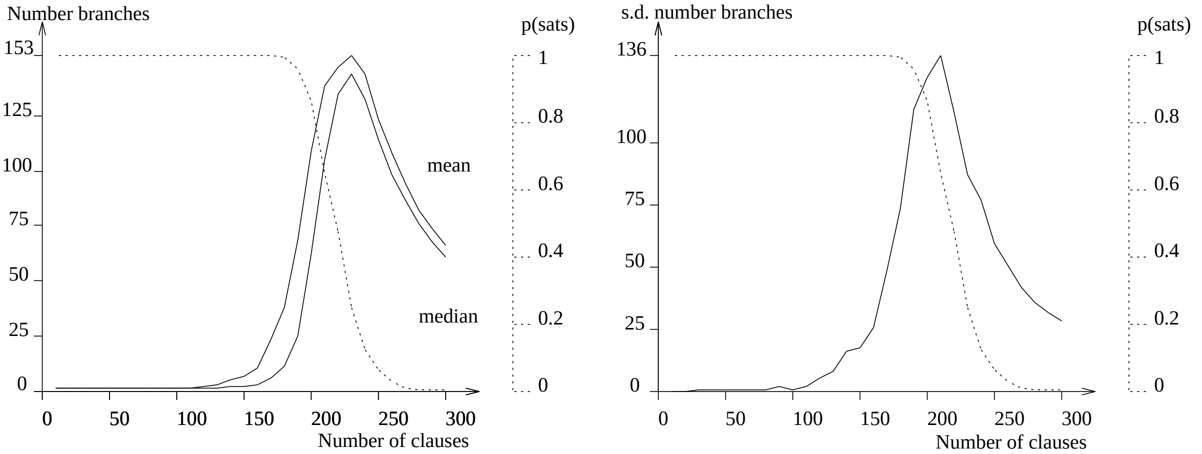
3 The Davis Putnam Algorithm

A standard procedure for determining satisfiability is due to Davis and Putnam [4]. This is shown in Figure 3. To simplify a clause set with respect to a partial truth assignment, we delete each clause that is satisfied by the partial truth assignment, and in every other clause delete any literals that contradict the partial truth assignment. Note that an empty clause set is trivially satisfiable, and a clause set containing an empty clause is trivially unsatisfiable.

The Davis-Putnam procedure is sound and complete: it returns "satisfiable" if and only if the clause set Σ is satisfiable. This property is maintained even if the tautology rule, unit propagation, and pure-literal deletion are omitted. For example, in their study of random 3-SAT, [13] used only unit-propagation and the split-rule. We will consider both this variant (which we will call "unit DP") as well as the full algorithm (which we will call "DP").

The Davis-Putnam procedure is non-deterministic since the literal used by the split-rule is unspecified.[1] We will first present results on implementations in which the split-rule simply chooses the first literal in the first clause in the clause set. However, heuristic procedures for determining a good literal to split on have been proposed so we also report results using the Jeroslow-Wang heuristic [10]. This

[1]The conditional part of the split-rule assigns a truth value to a particular variable v. If this variable is set to true, then we say that the split-rule uses the literal v. If this variable is set to false, then we say that it uses the literal $\neg v$.



(a) prob(sats), mean, median branches (b) prob(sats) and s.d. branches

Figure 2: unit DP, N=50, random 3-SAT problems[2]

heuristic estimates the contribution each literal is likely to make to satisfying the clause set. Each literal is scored as follows: for each clause c the literal appears in, $2^{-|c|}$ is added to the literal's score, where $|c|$ is the number of literals in c. The split-rule is applied to the literal with the highest score.

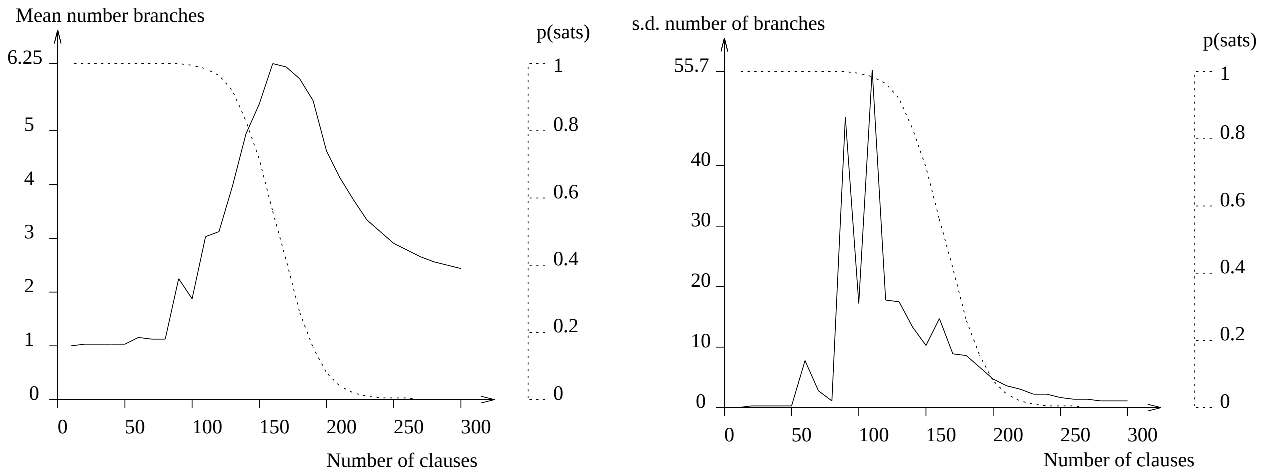
[13] have shown experimentally that the hardest random 3-SAT problems for unit DP are generated where the probability of satisfiability is around 50%, and that this happens where the number of clauses per variable is around 4.2 to 4.3. This result has been confirmed experimentally, by other workers [12, 3], using other procedures for SAT and up to much larger numbers of variables. We have observed similar behaviour. Figure 2a shows the mean and median number of branches[3] explored by unit DP plotted against the number of clauses for 50 variable random 3-SAT problems. The number of clauses goes from 10 to 300 in steps of 10. Each data point is the average over 1000 problems. Additionally, the dotted line indicates the probability of satisfiability. It is interesting to compare the mean and median in Figure 2a. The two graphs are very similar, with the median being slightly smaller than the mean, but of the same order. In Figure 2b we also plot the standard deviation in the number of branches over the same experiments. Note that the standard deviation is less than the mean in the difficult region, but slightly more in the easy region around 150 clauses. We return to a comparison of the mean and median for 3-SAT experiments later.

4 Hard Problems in CP

A number of theoretical results are known for the constant probability model showing that certain methods of varying L and p in terms of N give an average

[2]All graphs in this paper are linearly scaled by the maximal y-ordinate.

[3]By the number of branches we mean the number of leaf nodes in the search tree, or equivalently one more than the number of applications of the split rule.



(a) prob(sats) and mean branches (b) prob(sats) and s.d. branches

Figure 3: unit DP, N=50, random CP problems

running time polynomial in N. See for example [7, 6, 11, 5]. In some cases this might be because the problem classes do not cover a phase transition between satisfiable and unsatisfiable problems. Naturally, problem classes which are soluble in polynomial time on average, are unlikely to give hard instances of an NP-complete problem. In our experiments, we therefore varied L over a large enough interval that the phase transition was guaranteed to be traversed completely. To aid comparison with results for random 3-SAT, p was set to 3/200 at N=100, giving an average clause length of 3 (before the exclusion of unit and empty clauses). For other values of N, p was varied as lnN/N using the equation,[4]

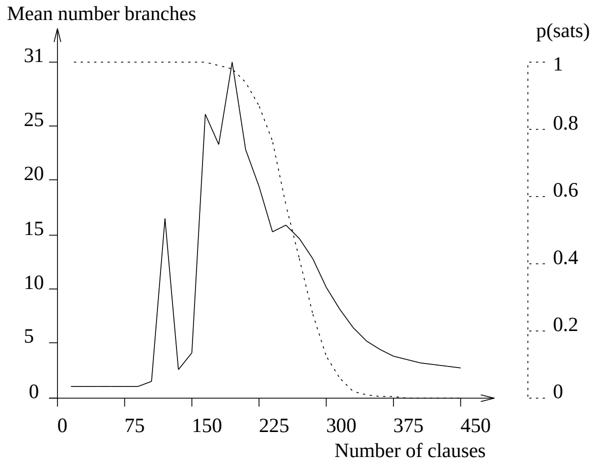
$$p = \frac{3}{200}\frac{\ln \mathrm{N}}{\mathrm{N}}\frac{100}{\ln 100}.$$

These parameter give problems which appear not to be soluble in polynomial average time as N increases.

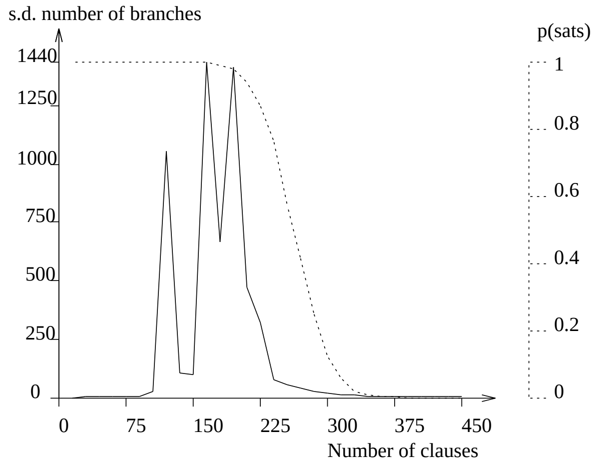
We first consider experiments with 50 variable CP problems: these are thus comparable in size to the 3-SAT data presented in Figures 2. In this experiment, each data point represents the result of 5000 experiments. Data points are plotted at intervals of 10 clauses. In general in this paper, we perform experiments with intervals in number of clauses of 0.2N, where N is the number of variables. Although this is rather coarse grained, and finer detail is desirable over the most interesting regions, this approach makes it feasible to perform large number of experiments at each point while covering a large range of parameter settings. Results are shown in Figure 3.

There are two immediate points about Figure 3a. First, it shows the standard easy-hard-easy pattern as the number of clauses increases, and the hard region is associated with the crossover from satisfiable to unsatisfiable problems. Second, the peak average number of branches is only 6, many times less than the peak in

[4]We thank Paul Purdom for suggesting this scaling for p.



(a) prob(sats) and mean branches



(b) prob(sats) and s.d. branches

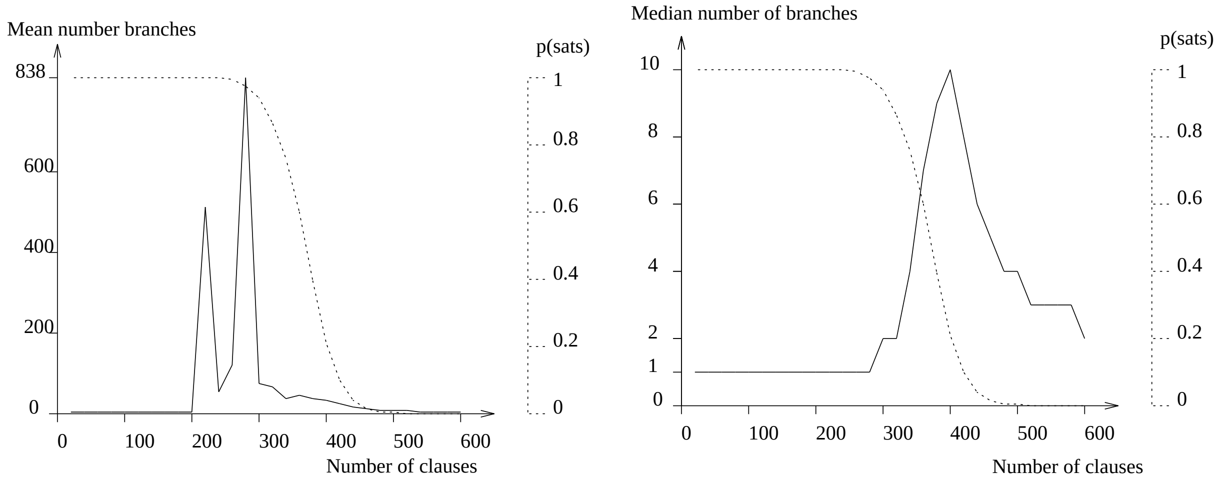
Figure 4: unit DP, N=75, random CP problems

Figure 2a. Given that the constant probability model is considered to generate easy problems [13], this is as expected. Another interesting feature of the data, though, is the apparent noise in the graph from about 75 to 100 clauses, despite the fact that 5000 experiments were performed at these points. The explanation for this becomes quite clear when the standard deviation of these experiments is plotted in Figure 3b.

The observed peak in the standard deviation of 56 is more than 15 times the average number of branches at that point, and nearly 10 times the maximum of the average number of branches for the whole graph. At the maximum average number of branches, at 160 clauses, the standard deviation is about 14, only twice the average. This behaviour is seen to a much milder degree in Figure 2 where the s.d. is about three times the average at 140 clauses, while it is lower than the average in the difficult region. Also, Figure 2a shows that this was the region with the largest discrepancy between mean and median.

Figure 4a shows the average number of branches explored at N=75 for the CP model. Again each data point is the result of 5000 experiments. The maximum average number of branches is now *not* seen in the region where approximately 50% of problems are satisfiable, but in a region where a very high percentage are satisfiable. The maximum average number of branches is still considerably less than the maximum of Figure 2a although that experiment was on smaller problems. There is also a local maximum in the region of 50% satisfiability. However, it is not clear whether this local maximum is genuine or is accounted for by noise in the data. Figure 4b shows the standard deviation on these experiments. The maximum standard deviation is now approximately 50 times the maximum average number of branches.

At N=100 variables, we performed only 1000 experiments at each data point due to limits on computational resources. Despite this, one satisfiable problem with 260 clauses, in a data set of which 99.4% were satisfiable, required 350,163,776



(a) prob(sats) and mean branches

(b) prob(sats) and median branches

Figure 5: unit DP, N=100, random CP problems

branches to solve.[5] This was considerably more than the total number of branches required by unit DP on a *thousand* 100 variable problems from the hard region of random 3-SAT.[6]

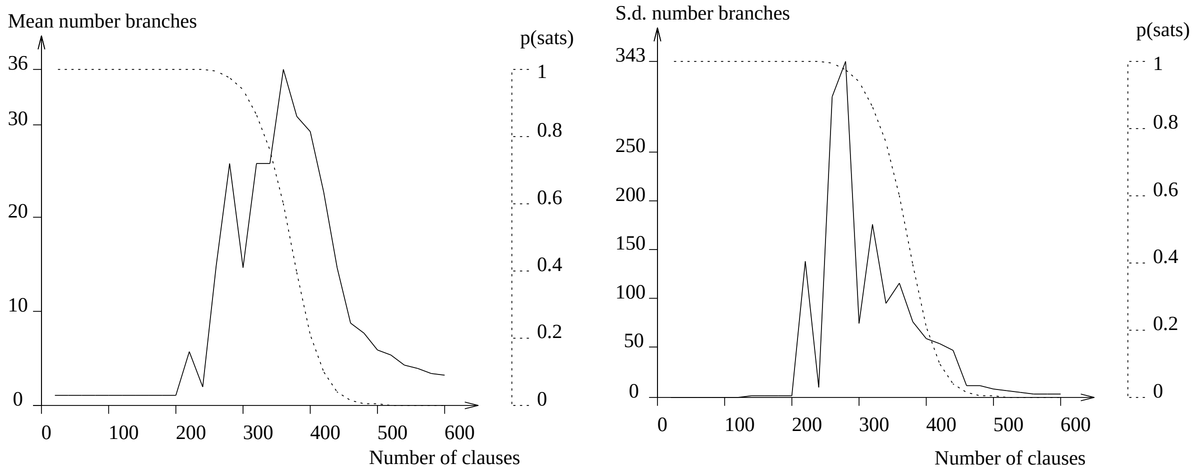
Even ignoring the incredible difficulty of this one problem, the effect we have been describing becomes much more pronounced than it was at 75 variables. Figure 5 shows the mean number of branches needed with the exceptional problem removed from the relevant data set. The high percentage satisfiability region totally dominates the graph. Note that there is a small local maximum at 360 clauses, where we would expect it in the usual "hard" region. Needless to say, standard deviations are enormous in the high satisfiability region.

It is interesting to compare Figure 5a with Figure 5b, in which the median rather than mean behaviour is plotted. The median behaviour follows a simple "easy-hard-easy" pattern, with the hard region coinciding with a region of approximately 50% satisfiability. Compared to random k-SAT, the maximum median reaches only a very small number of branches even in the difficult region. This picture is entirely different from the mean behaviour.

To summarise, we have shown that there is an incredible variability in the number of branches needed to solve problems generated randomly from the constant probability model without empty and unit clauses, using the Davis-Putnam procedure with unit propagation only. This variability is sufficient to affect mean behaviour by many orders of magnitude. The behaviour is observed in regions where more than 90% of problems are satisfiable. We have also observed a more

[5]On a 30 Mips machine this one problem took a Common Lisp implementation of unit DP about a week of CPU time.

[6]Compare also statistics reported by [15] for unit DP. For 100 variable random 3-SAT problems, in the difficult region where $L = 4.3\text{N}$ they report an average 84,000 recursive calls to their procedure. In solving this problem our procedure made more than 1.8 billion recursive calls to itself in exploring its 350 million branches.



(a) prob(sats) and mean branches (b) prob(sats) and s.d. branches

Figure 6: DP, N=100, random CP problems

normal "easy-hard-easy" pattern in the median number of branches used, but with a very much lower maximum than in random k-SAT problems of the same size, in line with data presented in [13].

5 Hard Problems for Better Algorithms

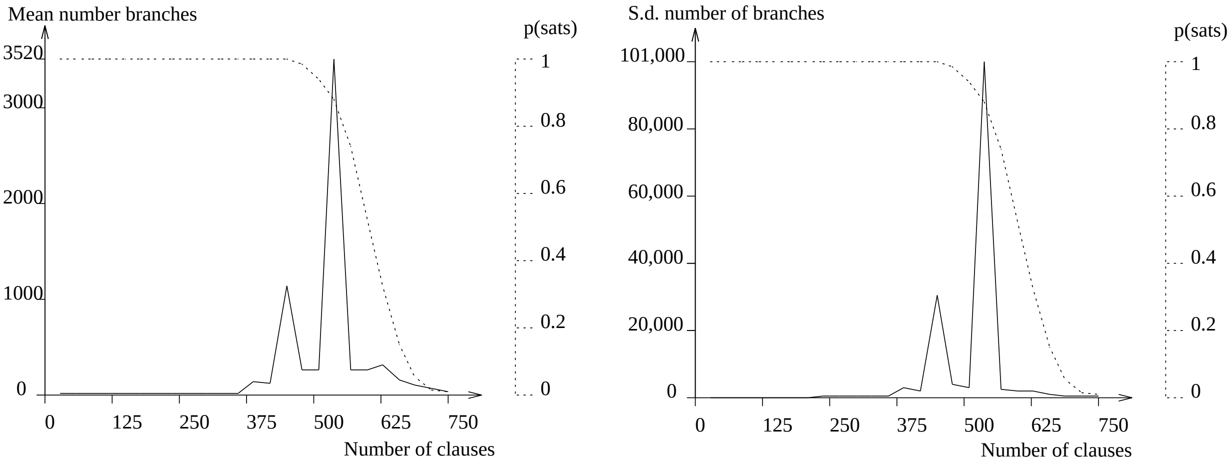
It might be thought that use of the full Davis-Putnam algorithm might eliminate this variability. Failing that, perhaps heuristics for the choice of literals in the split rule might eliminate this behaviour. However, this does not seem to be possible, as we show in this section.

First, we consider the full unsimplified Davis-Putnam algorithm, DP.[7] Figure 6 shows results for DP applied to the same sets of 100-variable CP problems used in the previous section. As a point of methodology, all experiments used the same problem sets to reduce variance between results.

Comparison of Figure 6a with Figure 5a shows that mean behaviour is very much better with DP than with unit DP. The same very large maxima in average number of branches are no longer observed. However, there is still sufficient variability in the data (despite 1000 experiments at each point) to produce considerable noise in the graph, and this is quite clear when the standard deviation is considered (Figure 6b). This behaviour is qualitatively similar to that of unit DP on 50 variable problems (Figure 3).

Figure 7 shows the results of DP applied to 150 variable CP problems, with 1000 experiments performed at each point. Behaviour is very similar to that of unit DP at 100 variables (Figure 5). The most difficult problem involved search of 3,197,178 branches to solve a 540 clause problem, in a problem set where 87.7% of

[7]This answers one objection the reader may have thought of, namely that we allow tautologies in our clause sets but (foolishly?) failed to filter them out in the simplified algorithm.



(a) prob(sats) and mean branches

(b) prob(sats) and s.d. branches

Figure 7: DP, N=150, random CP problems

problems were satisfiable.[8] This shows clearly that the enormous variability and resulting huge run-times for individual problems occurs with DP just as it does for the simpler unit DP. However, it seems that the increased sophistication of the full algorithm allows the worst behaviour to be postponed somewhat.

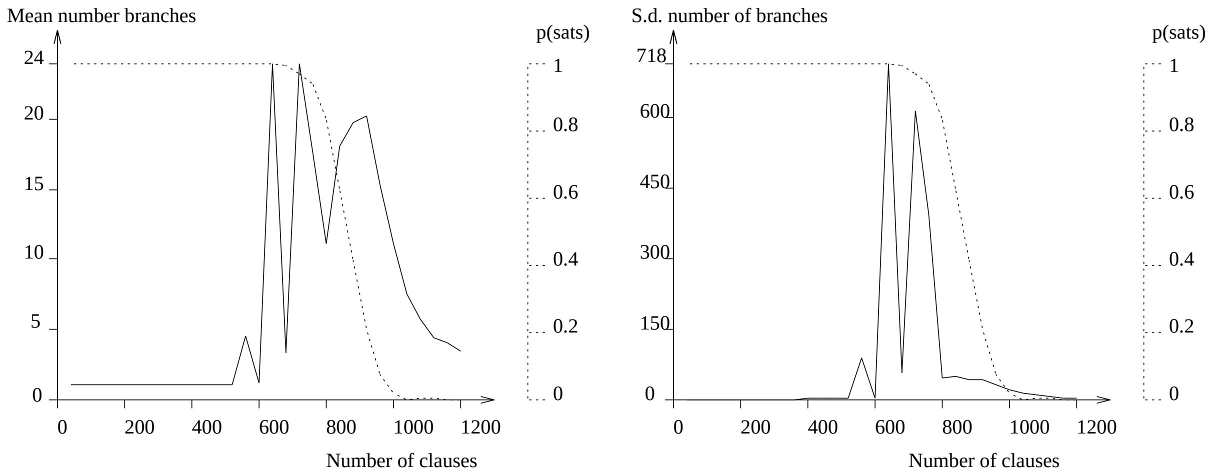
The versions of the Davis-Putnam procedure used in these experiments take a very naive approach to application of the split-rule, simply splitting on the first literal in the first clause. Heuristics can obviously be used to decide appropriate literals for splitting. One such heuristic is the Jeroslow-Wang heuristic described in §3.

Even using such sophisticated heuristics as Jeroslow-Wang, we cannot eliminate variability. Figure 8 shows the results of applying DP with this heuristic to 200 variable CP problems, with 1000 problems tested at each data point. This clearly shows that the worst mean behaviour is in a region of high percentage satisfiability, and that very high standard deviations are observed in that region. The most difficult problem we observed contained 640 clauses, and searched 22,722 branches, in a problem set of which 99.7% were satisfiable. Although it is clear that the Jeroslow-Wang heuristic does offer great improvements over the naive splitting heuristic, it still seems impossible to avoid highly variable and bad worst case behaviour.

Finally, it is interesting to note that despite the highly variable behaviour exhibited by both procedures examined in this section, the median behaviour of both is very much easier than mean behaviour, and exhibits a simple easy-hard-easy pattern observed with unit DP in Figure 5b. Figure 9 shows the median behaviour of DP at 150 variables, and of DP with Jeroslow-Wang branching at 200 variables.

To summarise this section, we have shown that the enormous variability in problem difficulty seen in §4 also occurs in the full Davis-Putnam procedure both

[8] This problem needed more than $2\frac{1}{2}$ days of CPU time on a 30 Mips machine.



(a) prob(sats) and mean branches (b) prob(sats) and s.d. branches

Figure 8: DP with Jeroslow-Wang, N=200, random CP problems

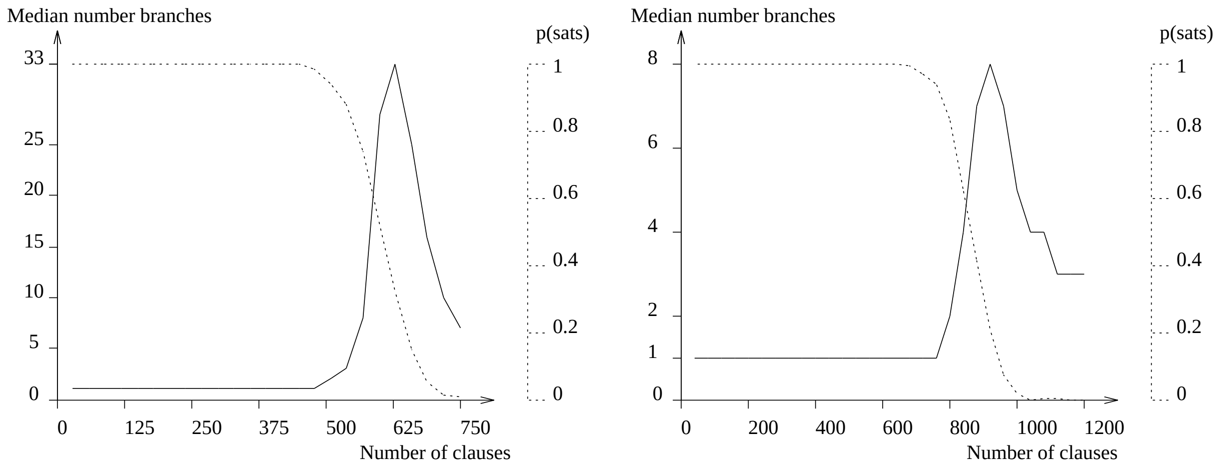
with and without heuristic choice of branching. However, the use of increasingly sophisticated procedures seems to delay the onset of the worst effects of this variability. We have also shown that when median behaviour is considered instead of mean behaviour, the full Davis-Putnam procedure exhibits an easy-hard-easy pattern associated with the transition from satisfiable to unsatisfiable problems, and that even in the hardest region the median problem requires very little search.

6 Problem Presentation

The constant probability model can generate enormously hard problems for backtracking algorithms in regions of otherwise easy problems. To understand this phenomenon more deeply, we present some analysis, and the results of some experiments, which start to provide an explanation.

It might be argued that the constant probability model has a finite probability of generating any given problem, and that perhaps by an extreme stroke of luck we happen to be observing some of the most difficult possible problems. There are two reasons why this argument fails. First, the variability increases massively with N. If these difficult problems were very unlikely we would expect them to appear with vanishingly small probability as N increases. Second, it seems that the problems which cause enormous difficulty are not necessarily *fundamentally* difficult. To explore the inherent difficulty or otherwise of the problem described in §4 that required 350 million branches, we gave 1000 random permutations, of this problem to the DP and unit DP procedures.[9] The results are given in Table 1.

[9]There are four ways to permute a set of clauses randomly without changing it in any way logically: the clauses can be permuted randomly; the literals in each clause can be permuted randomly; the names of the literals can be permuted randomly; and, after a coin toss, a literal can be swapped for its negation throughout the clause set. Not all of these will necessarily affect a given procedure.



(a) DP, N=150

(b) DP with Jeroslow-Wang, N=200

Figure 9: Median behaviour, random CP problems

Procedure	Median	Mean	s.d.	Worst case
unit DP	1	63,956	1,031,026	24,506,182
DP	1	116	1,321	26,245
DP + JW	1	1	0	1

Table 1: Performance on randomisations of a difficult problem.

More than half the time no search at all was required by unit DP to solve a problem which at worse needed a week's worth of search. Indeed, only 22 of the random permutations required more than the mean number of branches. Similar variable behaviour was observed with DP. This highly variable behaviour is similar to that observed on the problem set as a whole, except more pronounced since a much larger proportion of the randomised problem set required a large number of branches. The Jeroslow-Wang heuristic seems good enough to overcome this variability, so this problem does not seem to be inherently difficult.

The particular problem under investigation was satisfiable. An interesting possibility is that early on a bad split is made, leading to a very difficult unsatisfiable problem.[10] To test this conjecture, we examined the problem in detail, and discovered that when simplified on the first literal in the first clause, it was unsatisfiable. Inevitably then, both DP and unit DP will simplify on this literal, search until they prove the resulting clause set is unsatisfiable, then simplify on the negation of the literal. To our surprise, when simplified on the negation of the first literal, unit DP required only one branch to solve the problem.

Having seen that an initial bad choice can produce a very difficult and unsatisfiable problem, we examined how inherently difficult the resulting problem was by testing 100 randomisations of the reduced problem. Results are shown in Table 2.

[10]We thank Alan Bundy for suggesting this to us.

Procedure	Median	Mean	s.d.	Worst
DP	1,386	5,149	9,042	59,443
DP + JW	298	297	30	373

Table 2: Performance on randomisations of reduced problem.

Unfortunately, the problem was too difficult to test unit DP extensively. There is a very large increase in median difficulty for DP with and without Jeroslow-Wang branching. Note that there is now a small amount of variability even with this branching heuristic. Comparison with Figure 9b suggests that this problem is orders of magnitude harder than most problems of even 200 variables. This strongly suggests that once one bad choice is made, an inherently hard problem results.

To conclude, these results suggest that the variable behaviour described in §4 and §5 might be explained as follows. At small number of clauses where most problems generated are satisfiable, those problems generated that are unsatisfiable are sometimes very hard. In addition, some problems are generated which are satisfiable but reduce to a fundamentally hard unsatisfiable problem if the wrong reduction is applied at an early stage.

7 Related Work

Various theoretical results have been shown for the constant probability model without the deletion of empty and unit clauses. For example, Purdom and Brown [11] show that average running time is polynomial if any of the following hold asymptotically: (1) $\mathrm{L(N)} \leq \mathrm{M}\,ln\mathrm{N}$ for some constant M; (2) $\mathrm{L(N)} \geq e^{\epsilon \mathrm{N}}$ for some constant ϵ; (3) $p(\mathrm{N}) \geq \epsilon$; or (4) $p(\mathrm{N}) \leq \mathrm{M}(ln\mathrm{N}/\mathrm{N})^{3/2}$. It is important to note, however, that these results do not cover all possible functions $p(\mathrm{N})$ and L(N), and do not in the limit cover the phase transition between satisfiable and unsatisfiable.

In an empirical study of random 3-SAT, Crawford and Auton have observed secondary peaks in problem difficulty in regions of high satisfiability, and also write that "below the cross-over point ...there are some problems that are as hard as those at the cross-over point" [3]. They do not attempt to explore this phenomenon further or to provide an explanation. Their results, however, seen in the light of work presented here, suggest strongly that the behaviour we have seen in the constant probability model will be duplicated in 3-SAT.

In an earlier empirical study of random 3-SAT, Mitchell *et al* [13] may have observed similar behaviour to Crawford and Auton as they note: "The means of the numbers of calls are influenced by a very small number of very large values. As the median is less sensitive to such 'out-liers', it appears to be a more informative statistic." Unfortunately, they fail to give details of any variable behaviour they may have observed. Our results suggest that the median is not the most informative statistic. Sensible estimates for the run time of a large series of experiments must be based on the mean and standard deviation, not the median.

Williams and Hogg have developed a simplified model for the difficulty of constraint satisfaction problems like SAT based upon an abstract lattice of possible variable assignments [16]. They predict that "Below the transition region ... a few very difficult cases raise the mean [above the median]. This behaviour, which becomes more extreme as μ [number of variables] increases, gives another contribution to the difference between our cost proxy and the actual cost observed" (p.24 of [16]). Our results suggest that this difference cannot be discounted.

It appears that a number of researchers may have observed variable behaviour like that presented in this paper, but its importance does not seem to have been realised.

8 Conclusions

The "easy" region of the constant probability model can sometimes give problems much harder for the Davis-Putnam procedure than the hardest problems of the "hard" region of the random k-SAT model. These very hard problems occur in the phase transition between satisfiable and unsatisfiable problems where there is at least a 90% chance of problems being satisfiable (and not 50% as had been previously proposed). Although such hard problems are rare, they increase in frequency and difficulty with N, the number of variables. The use of better heuristics appears to delay but not postpone indefinitely the appearance of these difficult problems. This is result of fundamental importance to experimental AI, since the comparison of algorithms for NP-complete problems requires the identification of regions in which hard problems are generated.

We have also observed an easy-hard-easy pattern in the median problem difficulty for the constant probability model, peaking where about 50% of problems are satisfiable (see also [13, 9]). The mean reaches a local maximum at the 50% point but has much smaller variability than in the 90% region.

The hardest problems appear to be either fundamentally hard unsatisfiable problems or satisfiable problems which contain very hard unsatisfiable subproblems. We suggest that these difficult unsatisfiable problems (or subproblems) are "critically constrained". That is, despite having relatively few clauses and being in a region where most problems are satisfiable, there are just enough constraints upon the variables for the problem to be unsatisfiable. It is thus very difficult for their unsatisfiability to be determined without an exhaustive and exponential search.

We have demonstrated quite convincingly that the most difficult problems in a particular problem class for a particular class of algorithms for SAT lie away from the hardest median behaviour, and can be orders of magnitude harder than worst case median behaviour. We conjecture that this phenomenon will occur for many other, and possibly all, algorithms for SAT. Furthermore, it is also likely that similar phenomena will be observed, not only in other classes of randomly generated SAT problems, but also in other NP-hard problems. The hardness of many NP-hard problem classes may lie with rare but exponentially hard problems that are found in otherwise easy regions.

Acknowledgements

This research was supported by a SERC Postdoctoral Fellowship to the first author and a HCM Postdoctoral fellowship to the second. We thank Paul Purdom for his advice, and Alan Bundy, and the members of the Mathematical Reasoning Group for their constructive comments and trillions of their CPU cycles.

References

[1] P. Cheeseman, B. Kanefsky, and W.M. Taylor. Where the really hard problems are. In *Proceedings of the 12th IJCAI*, pages 163–169. International Joint Conference on Artificial Intelligence, 1991.

[2] S.A. Cook. The complexity of theorem proving procedures. In *Proceedings of the 3rd Annual ACM Symposium on the Theory of Computation*, pages 151–158, 1971.

[3] J.M. Crawford and L.D. Auton. Experimental Results on the Cross-Over Point in Satisfiability Problems. In *Proceedings of AAAI 1993 Spring Symposium on AI and NP-Hard Problems*, 1993.

[4] M. Davis and H. Putnam. A computing procedure for quantification theory. *J. Association for Computing Machinery*, 7:201–215, 1960.

[5] John Franco and Yuan Chuan Ho. Probabilistic performance of a heuristic for the satisfiability problem. *Discrete Applied Mathematics*, 22:35–51, 1988.

[6] John Franco and Marvin Paull. Probabilistic analysis of the davis-putnam procedure for solving the satisfiability problem. *Discrete Applied Mathematics*, 5:77–87, 1983.

[7] Allen Goldberg. Average case complexity of the satisfiability problem. In William H Joyner Jr., editor, *Proceedings, Fourth Workshop on Automated Deduction*, Austin, Texas, February 1–3 1979.

[8] J. N. Hooker. Solving the incremental satisfiability problem. *Journal of Logic Programming*, 15:177–186, 1993.

[9] J. N. Hooker and C. Fedjki. Branch-and-cut solution of inference problems in propositional logic. *Annals of Mathematics and Artificial Intelligence*, 1:123–139, 1990.

[10] R. E. Jeroslow and J. Wang. Solving propositional satisfiability problems. *Annals of Mathematics and Artificial Intelligence*, 1:167–187, 1990. (cited in [8]).

[11] Paul Walton Purdom Jr. and Cynthia A. Brown. The pure literal rule and polynomial average time. *SIAM Journal of Computing*, 14(4):943–953, November 1985.

[12] T. Larrabee and Y. Tsuji. Evidence for a Satisfiability Threshold for Random 3CNF Formulas. Technical Report UCSC-CRL-92-42, Baskin Center for Computer Engineering and Information Sciences, University of California, Santa Cruz, 1992.

[13] David Mitchell, Bart Selman, and Hector Levesque. Hard and easy distributions of SAT problems. In *AAAI-92: Proceedings Tenth National Conference on Artificial Intelligence*. AAAI Press/The MIT Press, July 12-16 1992.

[14] R. Reiter and A. Mackworth. A logical framework for depiction and image interpretation. *Artificial Intelligence*, 41(3):123–155, 1989.

[15] B. Selman, H. Levesque, and D. Mitchell. A New Method for Solving Hard Satisfiability Problems. In *Proceedings of the 10th National Conference on AI*, pages 440–446. American Association for Artificial Intelligence, 1992.

[16] C. Williams and T. Hogg. Exploiting the Deep Structure of Constraint Problems. PARC preprint P92-00005, Dynamics of Computation Group, Xerox Palo Alto Research Center, 1992.