# Polynomial Average-Time Satisfiability Problems

by

Paul Walton Purdom, Jr. and Cynthia A. Brown

## Abstract

A class of random conjunctive normal form (CNF) predicates is characterized by $v$, the number of variables from which literals may be formed (giving $2v$ literals); $p(v)$, the probability that a given literal is in a random clause; and $t(v)$, the number of random independently selected clauses in a random predicate. Determining satisfiability on such a class using backtracking takes polynomial average time under the following conditions: (1) $\lim_{v\to\infty} vp(v) = 0$ and $t(v) \geq \frac{(\ln 2+\epsilon)v}{-\ln((v+1)p(v))}$; and (2) $\lim vp(v) = \infty$, $\lim p(v) = 0$, and $t(v) \geq \dfrac{(\ln 2 + \epsilon)}{ep(v)} \exp[2vp(v)]$ for any fixed $\epsilon > 0$. Additional cases are also covered. An analysis by Goldberg showed that problems for which $p(v)$ is constant can be solved in polynomial average time by a *simplified Davis-Putnam procedure.* We give an algorithm for solving problems with $t(v) \leq \ln\ln v/\ln 3$ in polynomial worst-case time. Thus all problem sets for which $p(v)$ or $t(v)$ grows extremely rapidly or extremely slowly can be solved in polynomial average time. We also show that *backtracking* requires exponential average time when $\lim vp(v) = 0$ and $t(v) \leq \frac{(\ln 2-\epsilon)v}{-\ln((v+1)p(v))}$, and when $\lim vp(v) = \infty$, $\lim_{v\to\infty} vp(v) = 0$, and $t(v) \leq \frac{(\ln 2-\epsilon)}{ep(v)} \exp[2vp(v)]$.

This work was supported in part by The National Science Foundation under grant number MCS 7906110.

## §1 Introduction

The amount of time needed to solve NP complete problems is a major open question. Much theoretical work has been done on this problem; see Garey and Johnson [7] for a summary. Naive methods for solving NP complete problems with $v$ binary variables typically take time $2^v$. Recently, methods have been developed by Monien and Speckenmeyer and by Schoeppel and Shamir for solving some problems in worst-case time $c^v$, where $1 < c < 2$ [11,18].

While there is no known way to solve NP complete problems in subexponential worst-case time, most methods used in practice combine a worst-case time of $q(v)2^v$ (where $q(v)$ is a polynomial in $v$) with the ability to solve many problems rapidly. Evaluation and comparison of these methods hinges on an average time analysis of their performance. An average time analysis of *backtracking* over random satisfiability problems is given in [1,12]. An analysis of *simple search rearrangement backtracking* (equivalent to *backtracking* combined with the *pure literal rule* from the *Davis-Putnam procedure* [3]) using the model of satisfiability problems from [1] is given in [13]. The performance of the *pure literal rule* from the *Davis-Putnam procedure* was studied by Goldberg [8,9] using a different random model. A preliminary analysis of several algorithms on both types of models is given in [2]. Methods which take polynomial time and which nearly always find solutions were studied by Franco and Paull in [4] (based on *stopping at the first solution*) and [5,6] (based on the *pure literal rule*). Satisfiability of conjunctive normal form (CNF) predicates is a representative NP complete problem and it is straightforward to generate random problem instances, so it is a natural choice for a model of random NP complete problems.

The CNF predicates can be classified by the number of variables from which literals can be formed (if there are $v$ variables there are $2v$ literals), the probability $p$ that a given literal appears in a random clause, and the number $t$ of clauses. The studies mentioned above were based on specific models of CNF predicates obtained by parameterizing $p$ and $t$ in terms of $v$ and studying the behavior of the algorithm in question as $v$ becomes large. For this paper we use a very simple model of CNF predicates (adapted from [8]), but we generalize the analysis so that the probability $p(v)$ and the number of clauses $t(v)$ are arbitrary functions of $v$.

This work was motivated by the observation that the functional values of $p(v)$ and $t(v)$ for which backtracking can solve CNF predicates rapidly are quite different from those for which the pure literal rule is effective. This suggests a program of categorizing satisfiability algorithms according to the points in $[p(v), t(v)]$ space for which they can solve the corresponding problems in polynomial average time. It will be interesting to see whether there is some such algorithm for each set of values. Algorithms which can rapidly solve problems that no other analyzed algorithm handles efficiently are of particular interest.

This paper begins such a program. Figure 1 summarizes our results. Imagine a rectangular space with various functions $p(v)$ labelling a vertical axis along the left edge, and functions $t(v)$ labelling a horizontal axis along the bottom. The functions are arranged so that the most rapidly growing $t(v)$ are at the right and the most slowly decaying $p(v)$ are at the top. It should be pointed out that this quasi-graphical representation should not be taken too literally; the functions for $t(v)$ and $p(v)$ do not form a complete ordered field.

A model is represented by a point on this diagram. The formulas derived in this paper from the study of *backtracking* define a roughly diagonal line from top to bottom on this figure such that functions $t(v)$ to the right of and below the line result in polynomial average time while those to the left and above give exponential time. The points near the top of the diagram, where $p(v)$ is constant, represent problems that can be solved in polynomial average time using the *pure literal rule*, while those at the extreme left, where $t(v) \leq \ln\ln v/\ln 3$, can be solved in polynomial time using an algorithm presented in this paper. One interesting characteristic of a model is the expected number of solutions for a problem. If $t$ is slightly smaller than $(v \ln 2)/(-\ln(1-(1-p)^v))$ then the number of solutions per problem is exponentially large; if $t$ is slightly larger than this, the number of solutions is exponentially small. (See the Appendix for details.) The dashed line in Fig. 1 shows this critical value of $t$ . Points below and to the right of the dashed line (which coincides with the boundary of the hard problem region in part) have an exponentially small number of solutions; points above and to the left have an exponentially large number of solutions.

For purposes of this diagram, "time" means "number of major steps of the algorithm in question", measured as a function of $v$. Problems in the upper right corner have an exponential number of clauses, and the pure literal rule cannot solve such problems in an average number of steps polynomial in $v$. On the other hand, the problems can be solved in an average time polynomial in the problem size.

The remaining hard problems (those where none of the algorithms analyzed so far take polynomial average time) occupy a roughly triangular region in the center of the diagram. The aim of future work will be to further reduce the size of this central region. (See [15]).

## §2 The Probability Model

A model for average time analysis consists of a set of random problems and a probability for each problem in the set. For this study the problem set is satisfiability of random CNF predicates over $v$ variables. Random clauses are formed by selecting each literal independently, with probability $p(v)$. The probability that neither literal for a variable is included is $(1-p(v))^2$ and the probability that both literals are included is $p(v)^2$, so trivial clauses are common when $p(v) \gg v^{-\frac{1}{2}}$ and relatively rare when $p(v) \ll v^{-\frac{1}{2}}$ . Random predicates are formed by independently selecting $t(v)$ random clauses.

This method of forming predicates is similar to that used in [8,9]. In those models, however, the probability of neither variable occurring in a clause was set at $1-2p(v)$, and tautological clauses (clauses containing a variable and its negation) were excluded. The models in [1,4,6,13] use random clauses with a fixed number of literals per clause. The model used in [2] was the same as that used here. (In [2] we erroneously stated that this model was identical to the one in [8].)

The present model has the advantage of being easy to analyze, and the transition from polynomial time to exponential time as the parameters are varied is rapid. It probably gives results that are less indicative of the average time required for solving typical problems than the models used in [1,4], but the advantages of easy analysis and rapid transitions are particularly useful in our present effort to identify good features of algorithms and to characterize the hard problems.

To analyze backtracking it is necessary to specify not only the set of problems but also the predicates to be used for testing partial solutions (intermediate predicates). We use the convention of [1], where the intermediate predicate to use after the first $i$ variables have been set is the conjunction of the clauses in which only the first $i$ variables appear. We use the standard convention that an empty CNF predicate is *true* and an empty clause is *false*.

## §3 Exact Analysis of Backtracking

See [1 or 10] for a discussion of backtracking. We consider the version of backtracking that searches for all solutions to a problem. The *backtracking* algorithm, when used with binary variables, generates a binary tree. We use the number of nodes in the tree as a measure of the running time of the algorithm. The following derivation follows the pattern of the derivation in [1], where more details are given.

The probability that a random clause has all false literals after $i-1$ variables have been set is $(1-p(v))^{2v-i+1}$. The probability that a random predicate contains no clause consisting of all false literals (i.e., the probability it reaches the node on level $i$ on a particular branch of the tree) is $[1-(1-p(v))^{2v-i+1}]^{t(v)}$. There are $2^i$ nodes on level $i$, so the expected time (expected number of nodes) is

$$T_B = 1 + \sum_{1 \le i \le v} 2^i \left(1 - (1-p(v))^{2v-i+1}\right)^{t(v)}. \tag{1}$$

## §4 Asymptotic Analysis of Backtracking

An asymptotic analysis of (1) reveals its ultimate behavior. Since the analysis is long and complex, it is given in the appendix. Here we discuss a few aspects of the analysis and give the results.

The analysis has two cases depending on whether the maximum term in the sum in (1) is the $i = v$ term or a term with $i < v$. In the first case the asymptotic analysis depends on whether $\lim_{v\to\infty} vp(v) = 0$, $a$, or $\infty$, where $a$ is a positive constant. In the second case the results depend most directly on whether $\lim_{v\to\infty} t(v)p(v) = 0$, $a$, or $\infty$, but these cases turn out to be equivalent to the corresponding cases for $\lim_{v\to\infty} vp(v)$. The two cases lead to two curves, each of which has three segments. The curves intersect where $\lim_{v\to\infty} vp(v) = \ln 2$. If we regard the model problems, parameterized by $t(v)$ and $p(v)$, as forming a space (as in Figure 1), then when $\lim_{v\to\infty} vp(v)$ is below $\ln 2$ the curve from the $i = v$ case separates the regions of the space where backtracking uses exponential ($e^{\epsilon v}$) average time (the small $t(v)$ side) from regions where polynomial time is used. Above $\ln 2$ the curve for the $i < v$ case separates the two regions. In the limit the separating curve has four segments.

There are some functions, such as $(\ln v)^{\ln v}$, that represent a running time that is neither exponential nor polynomial under the above classification. In our diagram such functions represent a vanishingly small portion of the space, and we will not mention them again.

The results of the analysis are summarized below; in each case $\epsilon$ is a positive number.

**1.** For $\lim_{v \to \infty} vp(v) = 0$

$$\text{exponential time is used when } t(v) \le \frac{(\ln 2 - \epsilon)v}{-\ln((v+1)p)} \tag{2}$$

$$\text{polynomial time is used when } t(v) \ge \frac{(\ln 2 + \epsilon)v}{-\ln((v+1)p)} \tag{3}$$

**2.** For $\lim_{v \to \infty} vp(v) = a \le \ln 2$

$$\text{exponential time is used when } t(v) \le \frac{(\ln 2 - \epsilon)v}{-\ln(1 - e^{-a})} \tag{4}$$

$$\text{polynomial time is used when } t(v) \ge \frac{(\ln 2 + \epsilon)v}{-\ln(1 - e^{-a})} \tag{5}$$

**3.** For $\lim_{v \to \infty} vp(v) = b \ge \ln 2$

$$\text{exponential time is used when } t(v) \le \frac{(\ln 2 - \epsilon)vd}{b} \tag{6}$$

$$\text{polynomial time is used when } t(v) \ge \frac{(\ln 2 + \epsilon)vd}{b} \tag{7}$$

$$\text{where } d \text{ is the solution of } \ln(1+d) + d\ln(1 + 1/d) = 2b. \tag{8}$$

**4.** For $\lim_{v \to \infty} vp(v) = \infty$

$$\text{exponential time is used when } t(v) \le \tfrac{\ln 2(1-\epsilon)}{ep(v)} \exp[2vp(v)]. \tag{9}$$

If $\lim_{v \to \infty} p(v) = 0$ then

$$\text{polynomial time is used when } t(v) \ge \frac{\ln 2(1+\epsilon)}{ep(v)} \exp[2vp(v)]. \tag{10}$$

(See the appendix for the case $\lim_{v \to \infty} p(v) > 0$ .)

These results are illustrated in Figure 1, where the line dividing the exponential and polynomial regions has been sketched.

## §5 Other Algorithms

*Backtracking* is by no means the only algorithm that can solve some classes of satisfiability problems in polynomial average time. Goldberg [8,9] showed that when $p(v)$ is constant, a *simplified* version of the *Davis-Putnam procedure* that essentially uses only the pure literal rule solves the problems in the class in average time polynomial in the problem size. Although a different model was used in [8,9], the same result holds for the model used in this paper. The degree of the polynomial increases as the constant value of $p(v)$ becomes smaller. This allows us to draw a line across the top of Figure 1. The solid portion of the line shows where the average time is polynomial in $v$; the dashed portion shows where it is exponential in $v$ but polynomial in the problem size.

Another algorithm allows us to solve problems for which $t(v)$ is small (less than $\ln\ln v/\ln 3$) in polynomial worst-case time. We call it the *pattern of occurrences* algorithm. The *pattern of occurrence* of a variable $w$ is a vector with $t(v)$ components. The $i^{th}$ component is $+1$ if the positive literal $w$ appears in the $i^{th}$ clause, -1 if the negative literal $\neg w$ occurs, and 0 if neither literal occurs. (Tautological clauses are dropped in the first step of the algorithm.) We define the *opposite pattern of occurrence* to be the pattern of occurrence obtained by reversing the signs.

To use the *pattern of occurrence* algorithm first apply the following reduction:

1. Drop any tautological clauses.
2. Form the pattern of occurrences vector for each variable that occurs in the predicate.
3. If any two variables have the same pattern of occurrence, remove all the clauses containing them. (These clauses can be made true by setting one of the variables to true and the other to false, without affecting any other clauses.)
4. If any two variables have opposite patterns of occurrence, remove all clauses containing them. (The clauses can be made true by setting both variables to the same value, without affecting any other clauses.)
5. Drop any variables that don't occur in any of the remaining clauses.

After the reduction each variable has a unique pattern of occurrence, and no two variables have opposite patterns. The number of clauses remaining, $t'$, and the number of variables, $v'$, obey the relation

$$v' \leq \frac{3^{t'} - 1}{2} \leq \frac{3^t - 1}{2}.$$

Such problems can be solved by enumeration in time

$$2^{v'} \leq 2^{\frac{3^t-1}{2}}.$$

For polynomial time we must have, for fixed $\alpha$,

$$2^{\frac{3^t-1}{2}} \leq v^\alpha$$

$$\frac{(3^t - 1)}{2} \ln 2 \leq \alpha \ln v$$

$$3^t \leq \frac{2\alpha \ln v}{\ln 2} + 1$$

$$t \leq \frac{\ln(2\alpha \ln v + \ln 2) - \ln \ln 2}{\ln 3}.$$

So for

$$t \leq \frac{\ln \ln v}{\ln 3},$$

this method requires polynomial time. (Notice that this is a worst-case time, unlike our other results.)

The *pattern of occurrence* algorithm, and rules 6 and 7 of the algorithm of Monien and Speckenmeyer [11], are related to the *pure literal rule* of the *Davis-Putnam procedure* [3]. The *pure literal rule* applies to variables for which only one of the two possible literals actually occurs in the formula. If the variable is set to the value that makes that literal true, all the clauses in which it appears become true, regardless of the value of the other variables. The variable that occurs as a pure literal thus has a safe value. The idea of a *safe value* can be generalized to sets of variables, which may interact in such a way that they have safe values as a set, even though none of them has a safe value individually. Both the *pattern of occurrence* algorithm, and rules 6 and 7 of the algorithm of Monien and Speckenmeyer, identify special patterns of occurrences of variables that result in interacting safe values. A different generalization of the pure literal rule is given in [16].

## §6 Discussion

Formulas 2–10 demonstrate that *backtracking* can solve random CNF problems in polynomial average time when $t(v)$ is sufficiently large. The *pure literal rule* and related algorithms discussed in the previous section can solve problems where $t(v)$ or $p(v)$ is small. As Figure 1 illustrates, the regions covered by these algorithms include all the extreme values of $p(v)$ and $t(v)$.

The algorithms we analyzed for this paper were quite simple. It will be interesting to see how much the "island" of difficult problems can be reduced when more sophisticated algorithms are analyzed. It is also possible that combinations of techniques will be more effective than the individual techniques are when used alone. This is particularly likely to be the case when the cases that the techniques handle well individually are not too far apart. The algorithm of Monien and Speckenmeyer [11] shows that a judicious combination of the best currently known techniques can be significantly better in the worst case than simple approaches (worst-case time $O(1.62^v)$ versus $\Theta(2^v)$ for problems with 3 literals per clause).

Future work will concentrate on attempting to reduce the size of the regions of hard problems in $[p(v), t(v)]$ space. It seems unlikely that all the hard problems can be eliminated using presently available algorithms, but it will be interesting to see how far back the boundaries can be pushed.

Figure 2 summarizes the most recent results of our research, including work completed after that reported in this paper. The *useless variable* analysis is reported in [14]. The improved analysis of the *pure literal rule* is reported in [15]. A similar analysis of *search rearrangement backtracking* is reported in [17].

Figure 1. A diagram showing the regions of $[p(v), t(v)]$ space where random CNF predicates can be solved in polynominal average time. Problems outside the quadrilateral can be solved in polynomial average time. The various boundaries are labeled with the algorithm for producing polynomial average time in that region of the diagram. Along the *backtracking* boundary, we have shown that exponential average time is required (by backtracking) just inside the boundary. The other boundaries represent upper bound analyses only; they are therefore indicated using hash marks. The dashed line separates the region where problems have an exponentially large expected number of solutions (above the line) from the region where the expected number of solutions is exponentially small. Formulas on the diagram are approximations, simplified to show main features. We use $\epsilon$ for a very small positive constant. Along the vertical axis are the various functions for $p(v)$ , between 0 and 1, arranged by their asymptotic rate of growth. Along the horizontal axis are various functions for $t(v)$.

Figure 2. Figure 2 uses the same conventions as Fig. 1. The additional lines show the results of more recent upper bound analyses of the *pure literal rule* and a rule that discards *useless variables.* (See [15] and [14] respectively.)

# Appendix

This section gives the proofs of the formulas in the main text. The expected number of solutions per problem in our model is given by

$$S = 2^v(1 - (1 - p)^v)^t, \tag{A.1}$$

which can be rewritten as

$$S = e^{v \ln 2 + t \ln(1-(1-p)^v)}. \tag{A.2}$$

For $S = e^{\epsilon v}$, we have

$$\epsilon v = v \ln 2 + t \ln(1 - (1 - p)^v), \text{or} \tag{A.3}$$

$$t = \frac{v(\ln 2 - \epsilon)}{-\ln(1 - (1 - p)^v)}. \tag{A.4}$$

Notice that if $t$ is slightly smaller than $(v \ln 2)/(-\ln(1 - (1 - p)^v))$ then the number of solutions per problem is exponentially large; if $t$ is slightly larger than this, the number of solutions is exponentially small. In discussing asymptotic behavior, we use lim as an abbreviation for $\lim_{v \to \infty}$. The asymptotic behavior of this formula may be analyzed using results obtained later in this Appendix. The results of the rest of this paragraph depend on the rest of the Appendix through formula (A.27). In the case $\lim pv = 0$, we can apply (A.21) to (A.4) to obtain (A.24). In the case $\lim pv = a$ for some constant $a > 0$, apply (A.25) to (A.4) to obtain (A.27). In the case $\lim pv = \infty, \lim(1 - p)^v = 0$, so the number of terms needed so that the average number of solutions is $e^{\epsilon v}$ is given by

$$t = v(\ln 2 - \epsilon)(1 - p)^{-v}(1 + o(1)). \tag{A.5}$$

When $\lim pv = \infty$ and $\lim p = 0$, (A.5) can be simplified to

$$t = v(\ln 2 - \epsilon)e^{pv}(1 + o(1)). \tag{A.6}$$

In equations (A.3) - (A.6), when $\epsilon$ is close to $\ln 2$, almost all of the $2^v$ potential leaf nodes are solutions.

The expected time for backtracking on our model is given by

$$T_B = 1 + \sum_{1 \le i \le v} 2^i\left(1 - (1 - p)^{2v-i+1}\right)^t. \tag{A.7}$$

Notice that $T_B$ is a decreasing function of $t$ and an increasing function of $p$ and $v$. To simplify the notation we use $p$ and $t$ instead of $p(v)$ and $t(v)$ ; $p$ and $t$ are understood to be functions of $v$. We are interested in characterizing the functional values of $p$ and $t$ for which $T_B$ is subexponential for large $v$. Our first step is to find the value of the maximum term in the summation in (A.7). Define

$$T_B(i) = 2^i\left(1 - (1 - p)^{2v-i+1}\right)^t, \tag{A.8}$$

considered as a function of a real variable $i$ . Let $i_{\max}$ be the value of $i$ that maximizes $T_B(i)$, and let $i_*$ be the value of $i$ in the range $1 \le i \le v$ that maximizes $T_B(i)$ . Since

$T_B$ is greater than the largest term on the right side of (A.7), and less than $v$ times the largest term, we have

$$T_B = 1 + M(v)T_B(i_* + \delta) \tag{A.9}$$

where $1 \leq M(v) \leq v$ and $\delta$ is some number in the range $-1 < \delta < 1$; $\delta$ compensates for the fact that $i_*$ may not be an integer.

In what follows we will assume that $0 < p < 1$, and that $t \geq 1$.

To calculate $i_{max}$ we use

$$\left.\frac{d \ln T_B(i)}{di}\right|_{i_{max}} = 0 \tag{A.10}$$

(the natural log of $T_B(i)$ is easier to work with than the original formula). The derivative of $\ln T_B(i)$ at $i_{max}$ is given by

$$\ln 2 - \frac{t(1-p)^{2v-i_{\max}+1}(-\ln(1-p))}{1-(1-p)^{2v-i_{\max}+1}} = 0. \tag{A.11}$$

Solving for $(1-p)^{2v-i_{\max}+1}$, taking logarithms, and solving for $i_{\max}$ gives

$$i_{\max} = 2v + 1 - \frac{\ln[1 + \frac{t(-\ln(1-p))}{\ln 2}]}{-\ln(1-p)}. \tag{A.12}$$

Formula (A.11) shows that $\frac{d \ln T_B(i)}{di}$ is a decreasing function of $i$, so the place where the derivative goes to zero is a maximum.

Now, $i_* = \min(v, i_{\max})$. This is $i_{\max}$ when $v \geq i_{\max}$, i.e.

$$v \geq 2v + 1 - \frac{\ln[1 + \frac{t(-\ln(1-p))}{\ln 2}]}{-\ln(1-p)}. \tag{A.13}$$

Solving for $t$ gives

$$\ln\left[1 + \frac{t(-\ln(1-p))}{\ln 2}\right] \geq (v+1)(-\ln(1-p)), \text{ or} \tag{A.14}$$

$$t \geq \frac{\ln 2}{-\ln(1-p)}[\exp[(v+1)(-\ln(1-p))] - 1]. \tag{A.15}$$

When $t$ is below this limit, $i_* = v$ . We now derive a formula for $T_B$ using each of the two values of $i_*$ .

When $i_* = i_{\max}$ , (A.9) becomes

$$T_B = 1 + M(v)\exp\left[(\ln 2)(2v + 1 + \delta) - (\ln 2)\frac{\ln[1 + \frac{t(-\ln(1-p))}{\ln 2}]}{-\ln(1-p)} + t\ln\left[1 - (1-p)^{\left\{\ln\left[1+\frac{t(-\ln(1-p))}{\ln 2}\right]/(-\ln(1-p))\right\}-\delta}\right]\right] \tag{A.16}$$

Since

$$(1-p)^{\left\{\ln\left[1+\frac{t(-\ln(1-p))}{\ln 2}\right]/(-\ln(1-p))\right\}} = \frac{\ln 2}{\ln 2 + t(\ln(1-p))}, \tag{A.17}$$

formula (A.16) may be written as

$$T_B = 1 + M(v)\exp\left[(\ln 2)(2v+1+\delta) - \frac{\ln 2}{-\ln(1-p)}\ln\left[1+\frac{t(-\ln(1-p))}{\ln 2}\right] + t\ln\left(1-\frac{(\ln 2)(1-p)^{-\delta}}{\ln 2 + t(-\ln(1-p))}\right)\right]. \tag{A.18}$$

Formula (A.18) gives $T_B$ for the case $i_* = i_{\max}$. When $i_* = v$, (A.9) becomes

$$T_B = 1 + M(v)\exp\left[(\ln 2)v + t\ln[1-(1-p)^{v+1}]\right]. \tag{A.19}$$

We are interested in the asymptotic behavior of $T_B$ when $p$ and $t$ are functions of $v$. In particular, given $p(v)$, we wish to characterize the functions $t(v)$ that give $T_B$ the form

$$T_B = 1 + M(v)\exp(\epsilon v). \tag{A.20}$$

When $\epsilon = 0$, $T_B$ is polynomial; it is exponential for $\epsilon > 0$.

We divide the possible functional values for $p$ into cases depending on the value of $\lim v(-\ln(1-p))$. The values we consider are 0, a constant $a$ greater than zero, and $\infty$. To find the critical functional form for $t$ we equate (A.18) or (A.19) to (A.20) with $\epsilon = 0$ or a small positive number and solve for $t$.

The choice of whether to use (A.18) or (A.19) depends on the value of $\lim v(-\ln(1-p))$ under consideration; for each case one of the two formulas is appropriate. We present the calculations for the correct case; once an answer is obtained, the fact that $T_B$ is a monotone function of $t$ means that the case is unique.

Consider (A.19) when $\lim v(-\ln(1-p)) = 0$ (equivalent to $\lim pv = 0$). We use the assumption on the limit of $v(-\ln(1-p))$ to simplify (A.19). The derivation proceeds as follows. Equating the simplified expression and the functional form for a just barely exponential function (A.20) allows us to obtain the value of $t$ that makes $T_B$ exponential. The value we obtain for $t$ does not satisfy (A.15), so $i_{\max}$ is greater than $v$. Thus the derivation is consistent, since (A.19) is the correct formula for the case $i_* = v$. Now, $T_B$ is a monotone decreasing function of $t$. Since we have a value of $t$ that separates exponential from non-exponential behavior, that value must be unique.

We begin, then, with (A.19) and the assumption $\lim v(-\ln(1-p)) = 0$. Under the assumption we can write

$$\begin{aligned}(1-p)^{v+1} &= \exp[(v+1)\ln(1-p)] \\ &= 1-(v+1)p + O(vp^2).\end{aligned} \tag{A.21}$$

Substituting this in (A.19), we have

$$T_B = 1 + M(v)\exp[(\ln 2)v - t(-\ln((v+1)p))(1+O(p))] \tag{A.22}$$

For (A.22) and (A.20) to be true simultaneously, we have $\epsilon v = (\ln 2)v - t(-\ln((v+1)p)) + (1 + O(p))$, giving

$$t = \frac{(\ln 2 - \epsilon)v}{-\ln((v+1)p)(1 + O(p))} \tag{A.23}$$

$$t = \frac{(\ln 2 - \epsilon)v}{-\ln((v+1)p)}(1 + O(p)). \tag{A.24}$$

This value of $t$ fits the case $i_* = v$, since it satisfies (A.15). (That it does so can be seen from the analysis leading to (A.31)). We were therefore justified in using (A.19) to derive it.

Functions that satisfy (A.24) with $\epsilon = 0$ result in a polynomial value for $T_B$. Adding a small positive quantity to compensate for the $O$ terms gives that for $t \geq (\ln 2 + \epsilon)/-\ln((v+1)p)$, for any $\epsilon > 0$, $T_B$ is polynomial. For $t \leq \frac{(\ln 2 - \epsilon)v}{-\ln((v+1)p)}$, $T_B$ is exponential and, under our assumption that $\lim vp = 0$, the critical function for $t$ gives $\lim pt = \ln 2$. This completes the derivation of equations (2) and (3) in the text.

Now consider (A.19) when $\lim v(-\ln(1-p)) = a$, where $a$ is a constant greater than zero. (This assumption is equivalent to $\lim vp = a$. The general form of the argument is similar to the $\lim pv = 0$ case, as are the subsequent cases.) Using the assumption we have

$$\begin{aligned}(1-p)^{v+1} &= \exp[-(v+1)(-\ln(1-p))] \\ &= \exp(-a + o(1)) \\ &= e^{-a}(1 + o(1)).\end{aligned} \tag{A.25}$$

Using this in (A.19) we have

$$T_B = 1 + M(v)\exp\left[(\ln 2)v + t\ln(1 - e^{-a})(1 + o(1))\right]. \tag{A.26}$$

For (A.20) and (A.26) to hold simultaneously,

$$\epsilon v = (\ln 2)v + t\ln(1 - e^{-a})(1 + o(1)), \text{ or}$$

$$t = \frac{(\ln 2 - \epsilon)}{-\ln(1 - e^{-a})}v(1 + o(1)). \tag{A.27}$$

To have $\lim pv = a$ we must have $\lim p = 0$, and (A.15) becomes

$$t \geq \frac{\ln 2}{p}[\exp[a(1 + o(1))] - 1](1 + o(1)), \text{ or}$$

$$t \geq \frac{\ln 2}{p}(e^a - 1)(1 + o(1)). \tag{A.28}$$

Now, (A.27) comes from (A.19), which is the $i_* = v$ case. Formula (A.15), and therefore (A.28), is the condition for $i_* = i_{\max}$. We want $i_* = v$ (equivalent to reversing the inequality in (A.28)). Thus, for (A.27) to hold, we must have

$$\frac{(\ln 2-\epsilon)}{-\ln(1-e^{-a})}v \leq \frac{\ln 2}{p}(e^{a}-1)(1+o(1)). \qquad (A.29)$$

We are concerned with small $\epsilon > 0$ . Since $pv = a + o(1)$ ,

$$a \leq (e^{a}-1)(-\ln(1-e^{-a}))(1+o(1)). \qquad (A.30)$$

In the limit, (A.30) is satisfied by

$$a \leq \ln 2. \qquad (A.31)$$

Thus, for $a < \ln 2, T_B$ is polynomial for $t \geq \frac{\ln 2+\epsilon}{-\ln(1-e^{-a})}v$ and exponential for $t \leq \frac{\ln 2-\epsilon}{-\ln(1-e^{-a})}v$ . This completes the derivation of equations (4) and (5).

For the case of $\lim pv > \ln 2$, the result depends on the value of $\lim t(-\ln(1-p))$. Assume $\lim t(-\ln(1-p)) = a$, where $a$ is a constant, and consider formula (A.18), the $i_* = i_{\max}$ case. It gives

$$\begin{aligned} T_B &= 1 + M(v)\exp\left[(\ln 2)(2v+1+\delta) - \frac{\ln 2}{-\ln(1-p)}\ln\left[1+\frac{a}{\ln 2}+o(1)\right]\right.\\ &\quad \left. + t\ln\left[1-\frac{\ln 2(1-p)^{-\delta}}{\ln 2+a}+o(1)\right]\right]\\ &= 1 + M(v)\exp\left[(\ln 2)(2v+1+\delta) - t\frac{\ln 2}{a}\ln\left[1+\frac{a}{\ln 2}\right](1+o(1))\right.\\ &\quad \left. + t\ln\left[1-\frac{\ln 2(1-p)^{-\delta}}{\ln 2+a}\right](1+o(1))\right] \end{aligned} \qquad (A.32)$$

Satisfying both (A.20) and (A.32) requires that

$$\epsilon v = (\ln 2)(2v+1+\delta) - t\frac{\ln 2}{a}\ln\left(1+\frac{a}{\ln 2}\right) - t\ln\left(1-\frac{(\ln 2)(1-p)^{-\delta}}{\ln 2+a}\right)[1+o(1)], \text{ or}$$

$$t = \frac{(2\ln 2-\epsilon)v[1+o(1)]}{\frac{\ln 2}{a}\ln\left(1+\frac{a}{\ln 2}\right) - \ln\left(1-\frac{(\ln 2)(1-p)^{-\delta}}{\ln 2+a}\right)} \qquad (A.33)$$

For (A.33) to hold we have $\lim t = \infty$ (since $t$ is proportional to $v$), so for $\lim t(-\ln(1-p)) = a$, which is equivalent to $\lim pt = a$, we have $\lim p = 0$. We use this to replace $(1-p)^{-\delta}$ by 1 in (A.33), giving

$$t = \frac{(2\ln 2-\epsilon)v[1+o(1)]}{\frac{\ln 2}{a}\ln\left(1+\frac{a}{\ln 2}\right) + \ln\left(1+\frac{\ln 2}{a}\right)} \qquad (A.34)$$

Thus $T_B$ is polynomial for

$$t \geq \frac{(2\ln 2+\epsilon)v}{\frac{\ln 2}{a}\ln\left(1+\frac{a}{\ln 2}\right) + \ln\left(1+\frac{\ln 2}{a}\right)}$$

and exponential for

$$t \leq \frac{(2\ln 2 - \epsilon)v}{\frac{\ln 2}{a}\ln\left(1+\frac{a}{\ln 2}\right)+\ln\left(1+\frac{\ln 2}{a}\right)}.$$

When $\lim pt = a$ and $\lim pv = b \geq \ln 2$, we have $\lim \frac{t}{v} = \frac{a}{b}$. Using this in (A.34) with $\epsilon = 0$ gives a relation between $a$ and $b$:

$$\frac{a}{b} = \frac{(2\ln 2 + o(1))}{\frac{\ln 2}{a}\ln(1+\frac{a}{\ln 2})+\ln(1+\frac{\ln 2}{a})} \text{ or}$$

$$b = \left[\frac{1}{2}\ln\left(1+\frac{a}{\ln 2}\right)+\frac{a}{2\ln 2}\ln\left(1+\frac{\ln 2}{a}\right)\right][1+o(1)]. \qquad (A.35)$$

When $b = \ln 2$ this formula gives $a = \ln 2$; substituting this value in (A.27) and (A.34) shows that in the critical region $t$ is approximately equal to $v$. Since (4) (in the main text) is equivalent to (6) (and (5) to (7)) when $a = \ln 2$ and $b = \ln 2$ either can be used at this one point. This completes the derivation of equations (6), (7), and (8).

Now consider (A.18) when $\lim t(-\ln(1-p)) = \infty$ (which is equivalent to $\lim pt = \infty$). (We still have $\lim pv > \ln 2$.) First notice that

$$\begin{aligned}\ln\left[1+\frac{t(-\ln(1-p))}{\ln 2}\right] &= \ln\left[\frac{t(-\ln(1-p))}{\ln 2}\right]+\ln\left[1+\frac{\ln 2}{t(-\ln(1-p))}\right]\\ &= \ln\left[\frac{t(-\ln(1-p))}{\ln 2}\right]+O\left(\frac{1}{t(-\ln(1-p))}\right) \qquad (A.36)\end{aligned}$$

and

$$\begin{aligned}\ln\left[1-\frac{(\ln 2)(1-p)^{-\delta}}{\ln 2+t(-\ln(1-p))}\right] &= \ln\left[1-\frac{(\ln 2)(1-p)^{-\delta}}{t(-\ln(1-p))}\left(1+O\left(\frac{1}{t(-\ln(1-p))}\right)\right)\right]\\ &= -\frac{(\ln 2)(1-p)^{-\delta}}{t(-\ln(1-p))}\left[\left(1+O\left(\frac{1}{t(-\ln(1-p))}\right)\right)\right]. \qquad (A.37)\end{aligned}$$

Thus from (A.18) we get

$$\begin{aligned}T_B = 1 + M(v)\exp\Bigg[(\ln 2)(2v+1+\delta) - \frac{\ln 2}{-\ln(1-p)}\Bigg\{\ln\left[\frac{t(-\ln(1-p))}{\ln 2}\right]\\ + (1-p)^{-\delta} + O\left(\frac{1}{t(-\ln(1-p))}\right)\Bigg\}\Bigg]. \qquad (A.38)\end{aligned}$$

Satisfying both (A.20) and (A.38) requires that

$$\begin{aligned}\epsilon v =& (\ln 2)(2v+1+\delta)\\ &- \frac{\ln 2}{-\ln(1-p)}\left\{\ln\left[\frac{t(-\ln(1-p))}{\ln 2}\right]+(1-p)^{-\delta}+O\left(\frac{1}{t(-\ln(1-p))}\right)\right\},\end{aligned}$$

or

$$\ln\left[\frac{t(-\ln(1-p))}{\ln 2}\right] = \left(2-\frac{\epsilon}{\ln 2}\right)v(-\ln(1-p)) + (1+\delta)(-\ln(1-p)) - (1-p)^{-\delta} + O\left(\frac{1}{t(-\ln(1-p))}\right).$$

Solving for $t$ gives

$$t = \frac{(\ln 2)e^{-(1-p)^{-\delta}}}{(-\ln(1-p))(1-p)^{1+\delta}} \exp\left[\left(2-\frac{\epsilon}{\ln 2}\right)v(-\ln(1-p))\right]\left[1+O\left(\frac{1}{t(-\ln(1-p))}\right)\right]. \tag{A.39}$$

Multiplying through (A.39) by $(-\ln(1-p))$ shows that $\lim pt = \infty$ iff $\lim pv = \infty$ . Thus (A.39) implies that if $\lim pv = \infty$ then $T_B$ is polynomial for

$$t \geq \frac{(\ln 2)e^{-(1-p)}(1+\epsilon)}{(-\ln(1-p))(1-p)^2} \exp[2v(-\ln(1-p))]$$

and is exponential for $t \leq \frac{(\ln 2)e^{-(1-p)^{-1}}(1-\epsilon)}{(-\ln(1-p))} \exp[2v(-\ln(1-p))]$. When $\lim p = 0$ this can be simplified; $T_B$ is polynomial for $t \geq \frac{\ln 2(1+\epsilon)}{ep} \exp[2vp]$ and exponential for $t \leq \frac{\ln 2(1-\epsilon)}{ep} \exp[2vp]$ . This completes the derivation of equations (9) and (10) in the text.

Careful consideration shows that we have now taken care of all the cases that can arise; we have characterized the boundary between sets of formulas for which backtracking is exponential and those for which it is polynomial.

## References


1. Cynthia A. Brown and Paul Walton Purdom, Jr., "An Average Time Analysis of Backtracking", *SIAM J. Comp.* **10** (1981), pp. 583–593.

2. Cynthia A. Brown and Paul Walton Purdom, Jr., "How to Search Efficiently", *Proceedings of the Seventh Int. Joint Conf. on Artificial Intelligence,* **1** (1981), pp. 588–594.

3. Martin Davis and Hilary Putnam, "A Computing Procedure for Quantification Theory", *J. Assoc. Comput. Mach.* **7** (1960), pp. 201–215.

4. John Franco and Marvin Paull, "Probabilistic Analysis of the Davis Putnam Procedure for Solving the Satisfiability Problem", Case Institute of Technology Report No. CES–81–3 (June 1981).

5. John Franco and Marvin Paull, "Probabilistic Analysis of the Davis Putnam Procedure for Solving the Satisfiability Problem", *Discrete Applied Mathematics* **5** (1983), pp. 77-87.

6. John Franco, "Average Analysis of the Pure Literal Heuristic", Case Institute of Technology Report No. CES–81–4 (August 1981).

7. Michael R. Garey and David S. Johnson, *Computers and Intractability,* W. H. Freeman and Co., San Francisco (1979).

8. Allen Goldberg, "Average Case Complexity of the Satisfiability Problem", *Proceedings of the Fourth Workshop on Automated Deduction* (1970), pp. 1–6.

9. Allen Goldberg, Paul Walton Purdom, Jr., and Cynthia A. Brown, "Average Time Analyses of Simplified Davis-Putnam Procedures", *Information Processing Letters* **15** (1982), pp. 72–75.

10. Donald E. Knuth, "Estimating the Efficiency of Backtrack Programs", *Math. Comp.* **29** (1975), pp. 121–136.

11. Burkhard Monien and Ewald Speckenmeyer, "Three-Satisfiability is Testable in $0(1.62^r)$ Steps", Report No. 3, Theoretical Informatics Series, University of Paderborn (1979).

12. Bernard Nudel, "Consistent-Labeling Problems and their Algorithms: Expected-Complexities and Theory-Based Heuristics", *Artificial Intelligence* **21** (1983), pp. 135-178.

13. Paul Walton Purdom, Jr. and Cynthia A. Brown, "An Analysis of Backtracking with Search Rearrangement", Indiana University Computer Science Dept. Tech. Report No. 89 (1980), to appear in *SIAM J. Comp.*

14. Paul Walton Purdom, Jr. and Cynthia A. Brown, "Evaluating Search Methods Analytically", National Conference on Artificial Intelligence (1982), pp. 124–127.

15. Paul Walton Purdom, Jr. and Cynthia A. Brown, "The Pure Literal Rule and Polynomial Average Time", Indiana University Computer Science Dept. Tech. Report No. 128 (June 1982).

16. Paul Walton Purdom, Jr., "Solving Satisfiability Problems with Less Searching", Indiana University Computer Science Dept. Tech. Report No. 117 (1981).

**17.** Paul Walton Purdom, Jr., "Search Rearrangement Backtracking and Polynomial Average Time", *Artificial Intelligence* **21** (1983), pp. 117-133.

**18.** Richard Schoeppel and Adi Shamir, "A $T = 0(2^{n/2})$, $S = 0(2^{n/4})$ Algorithm for Certain NP Complete Problems", *SIAM J. Comp.* **10** (1981), pp. 456–464.