Theoretical Analysis of Davis-Putnam Procedure and Propositional Satisfiability

Nobuhiro Yugami
Fujitsu Laboratories
1015 Kamikodanaka Nakahara-ku
Kawasaki 211 Japan
E-mail: yugami@flab.fujitsu.co.jp

Abstract

This paper presents a statistical analysis of the Davis-Putnam procedure and propositional satisfiability problems (SAT). SAT has been researched in AI because of its strong relationship to automated reasoning and recently it is used as a benchmark problem of constraint satisfaction algorithms. The Davis-Putnam procedure is a well-known satisfiability checking algorithm based on tree search technique. In this paper, I analyze two average case complexities for the Davis-Putnam procedure, the complexity for satisfiability checking and the complexity for finding all solutions. I also discuss the probability of satisfiability. The complexities and the probability strongly depend on the distribution of formulas to be tested and I use the fixed clause length model as the distribution model. The result of the analysis coincides with the experimental result well.

1 Introduction

This paper presents a statistical analysis of the Davis-Putnam procedure [Davis and Putnam, 60] that solves propositional satisfiability problems (SAT). SAT is the problem to judge whether a given logical formula, given as a conjunctive normal form (cnf), can be satisfied or not. It has been researched in AI because of its strong relationship to automated reasoning such as theorem proving and planning. Recently, it is also concerned as a benchmark problem for constraint satisfaction algorithms [Selman, Levesque and Mitchell, 92][Yugami, Ohta and Hara, 94]. The Davis-Putnam procedure is one of the most famous algorithms for SAT.

A number of papers discussed the difficulty of SAT theoretically. Because of its NP-completeness, SAT requires more than polynomial time in the worst case, assuming NPnot =P, and the analysis of SAT concentrated on the average case behavior. Goldberg et. al. showed that the Davis-Putnam procedure, on average, could solve SAT in polynomial time of the number of variables [Goldberg, Purdom and Brown, 82]. They assumed the constant probability model as a distribution model of formulas to be tested. Franco and Paull pointed out that this easiness of SAT strongly depended on the distribution model and proposed a new distribution model, the fixed clause length model [Franco and Paull, 83][Franco, 86]. They showed that the Davis-Putnam procedure required exponential time for finding all solutions, assignments that satisfied the given formula, on this distribution model [Franco and Paull, 83][Franco, Plotkin and Rosenthal, 87]. This result is relevant to the original SAT, satisfiability checking, when few assignments satisfy the formula. Chao and Franco showed that the Davis-Putnam procedure with the unit clause rule could solve SAT without backtracking, i.e. it required only polynomial time on the fixed clause length model with a small clauses-to-variables ratio (smaller than 2.9 for 3-SAT). [Williams and Hogg, 94] analyzed the difficulty of general constraint satisfaction problems by using the structure of the assumption lattice. Recent experimental works showed more precise results of the average case behavior of the Davis-Putnam procedure. Mitchell et. al. reported the dependency of the Davis-Putnam procedure's average case complexity on the clauses-to-variables ratio and showed its drastic change near the 50% satisfiable point [Mitchell, Selman and Levesque, 92]. The complexity became maximum near the 50% satisfiable point and decreased rapidly when the ratio became larger or smaller. [Crawford and Auton, 93] reported similar results by using tableau based satisfiability checking algorithm.

The purpose of this paper is to analyze the Davis-Putnam procedure's average case behavior on the fixed clause length model, especially its dependency on the clauses-to-variables ratio. I discuss two complexities, the complexity for satisfiability checking and the complexity for finding all solutions. I also discuss the probability of satisfiability.

The organization of this paper is as follows. I first review the Davis-Putnam procedure and the fixed clause length model in the following two sections. Section 4 gives the statistical analysis of the average case complexities and the probability of satisfiability. The results of section 4 arc compared with experimental results in section 5. I summarize my results in section 6.

2 Davis-Putnam Procedure

Figure 1 shows the Davis-Putnam procedure (DP). It takes a cnf as an input and returns whether the cnf is "satisfiable" or "unsatisfiable" It solves SAT by searching an assignment that satisfies all clauses. DP uses the unit propagation (unit clause rule) to improve its performance. The unit propagation selects a variable in an unit clause, a clause containing a single literal, and assigns a value to the variable to satisfy the clause. In the original DP, the pure literal rule is also used to improve DP, but I ignore it in this paper because the unit propagation is much more powerful.

I use the number of DP calls as the complexity of DP. I modify DP to simplify the analysis. Figure 2 shows the modified procedure, SDP. The only difference between DP and SDP is that SDP does not check whether all clauses ate satisfied or not. It finds all clauses are satisfied when all variables are bound and no empty clause exists. The difference is not essential because if a partial assignment satisfies all clauses, then no backtrack occurs after that. This means that the number of SDP calls for satisfiability checking is larger than the number of DP calls by at most the number of variables and usually the difference is very small. For finding all solutions, there is no difference between DP and SDP.

3 The Fixed Clause Length Model

The complexity of SDP, and DP, strongly depends on the distribution of cnfs to be tested. It this paper, I use the fixed clause length model as the distribution model. This model has three parameters, the number of variables, *N,* the clause length, *K* and the number of clauses, *M.* A cnf on this model consists of *M* clauses and each clause contains *K* literals. In a clause, each variable appears at most once. A formula to be tested is randomly selected from the set of all cnfs that satisfy these conditions. The important feature of this model is that the clauses in the cnf are mutually independent. SAT on the fixed clause length model with clause length *K,* is calledk-SAT.

4 Analysis

4.1 Probabilities of becoming empty and unit

On k-SAT, a clause contains *K* literals first and becomes satisfied or becomes shorter when SDP assigns values to variables incrementally. In SDP, an empty clause and an unit clause perform important roles. I first discuss the probabilities of becoming empty or unit when *b* variables are bound and *r* variables are removed "Remove a variable"* means deleting both of positive and negative literals of the variable from the cnf. SDP, and DP, does not do such a thing but I will use removed variables to discuss the

```
procedure DP
    Input: A cnf to be tested
    Output: "satisfiable"or "unsatisfiable"
step 1 : Unit Propagation
    While there is no empty clause and an unit clause
    exists, select an unit clause and bind a variable in
    it to satisfy it.
step 2 : Satisfiability Checking
    If all clauses are satisfied, return "satisfiable".
step 3 : Unsatisfiability Checking
    If an empty clause exists, return "unsatisfiable"
step 4 : Splitting Rule
    Select a variable whose value is not assigned.
    Assign true to it and call DP. If the result is
    "satisfiable" then return "satisfiable". Otherwise,
    assign false to the variable and call DP again.
    Return the result of it.
```

Figure 1: Davis-Putnam Procedure

```
procedure SDP
    Input: A cnf to be tested
    Output: "satisfiable"or "unsatisfiable"
step 1 : Unit Propagation
    While there is no empty clause and an unit clause
    exists, select an unit clause and bind a variable in
    it to satisfy it.
step 2 : Unsatisfiability Checking
    If an empty clause exists, return "unsatisfiable"
step 3 : Satisfiability Checking
    If all variables are bound, return "satisfiable".
step 4 : Splitting Rule
    Select a variable whose value is not assigned.
    Assign true to it and call SDP. If the result is
    "satisfiable"then return "satisfiable". Otherwise,
    assign false to the variable and call SDP again.
    Return the result of it.
```

Figure 2: SDP, a variant of Davis-Putnam Procedure

probability of satisfiability. For simplicity, I use "at (b,r)" to represent "when *b* variables are bound and *r* variables are removed". Let$p_e(b,r)$ be the probability of becoming empty at *(b,r).* A clause becomes empty when all variables in it are bound or removed and values of bound variables violate the corresponding literals. Thus the desired probability is

$$p_e(b,r) = \sum_{n=\max(0,K-r)}^{\min(K,b)} \frac{\binom{b}{n}\binom{r}{K-n}}{\binom{N}{K}} \left(\frac{1}{2}\right)^n.$$

Where, n is the number of bound variables in the clause and $K - n$ variables in the clause are removed. The probability of becoming unit can be obtained by the same way. The only difference is that the clause contains one variable that is not bound If $b + r = N$ or $b + r \le K - 2$, the probability is 0 and otherwise,

$$p_u(b,r) = \sum_{n=\max(0,K-r-1)}^{\min(K-1,b)} \frac{\binom{b}{n}\binom{r}{K-n-1}\binom{N-b-r}{1}}{\binom{N}{K}} \left(\frac{1}{2}\right)^n .$$

Next, I discuss the transition probabilities from unit clause to empty clause, unit clause to unit clause and so on. An unit clause becomes empty if the variable of the remaining literal in it is bound and the assigned value violates the literal, or the variable is removed. Thus a clause that is unit at (b_1,r_1) becomes empty at (b_2,r_2) $(b_1 \le b_2, r_1 \le r_2)$ with probability

$$p_{u\to e}(b_1,r_1,b_2,r_2) = \frac{\frac{1}{2}(b_2-b_1)+(r_2-r_1)}{N-(b_1+r_1)} .$$

The probability that an unit clause at (b_1,r_1) is also unit at (b_2,r_2), $p_{u\to u}(b_1,r_1,b_2,r_2)$, and the probability of becoming satisfied at (b_2,r_2), $p_{u\to s}(b_1,r_1,b_2,r_2)$, can be calculated by the same way.

$$p_{u\to u}(b_1,r_1,b_2,r_2) = \frac{N-(b_2+r_2)}{N-(b_1+r_1)}$$

$$p_{u\to s}(b_1,r_1,b_2,r_2) = \frac{\frac{1}{2}(b_2-b_1)}{N-(b_1+r_1)}$$

Let $p_{o\to e}(b_1,r_1,b_2,r_2)$ be a probability that a clause, which is not empty nor unit at (b_1,r_1), becomes empty at (b_2,r_2). Because a clause is not empty nor unit at (b_1,r_1) with probability $1-p_e(b_1,r_1)-p_u(b_1,r_1)$,

$$p_e(b_2,r_2) = p_e(b_1,r_1) + p_u(b_1,r_1)p_{u\to e}(b_1,r_1,b_2,r_2) + \{1-p_e(b_1,r_1)-p_u(b_1,r_1)\}p_{o\to e}(b_1,r_1,b_2,r_2).$$

By solving this equation for $p_{o\to e}(b_1,r_1,b_2,r_2)$,

$$p_{o\to e}(b_1,r_1,b_2,r_2) = \frac{p_e(b_2,r_2)-p_e(b_1,r_1)-p_u(b_1,r_1)p_{u\to e}(b_1,r_1,b_2,r_2)}{1-p_e(b_1,r_1)-p_u(b_1,r_1)} .$$

Especially,

$$p_{o\to e}(b-1,0,b,0) = 0$$

because a value assignment to one variable can make a new empty clause only from an unit clause.

The probability of becoming unit can be calculated with the similar discussion.

$$p_{o\to u}(b_1,r_1,b_2,r_2) = \frac{p_u(b_2,r_2)-p_u(b_1,r_1)p_{u\to u}(b_1,r_1,b_2,r_2)}{1-p_e(b_1,r_1)-p_u(b_1,r_1)}$$

For simplicity, I will use the following notations.

$$p_{u\to e}(b) = p_{u\to e}(b-1,0,b,0)$$
$$p_{u\to u}(b) = p_{u\to u}(b-1,0,b,0)$$
$$p_{u\to s}(b) = p_{u\to s}(b-1,0,b,0)$$
$$p_{o\to u}(b) = p_{o\to u}(b-1,0,b,0)$$

4.2 Unit Propagation

The effectiveness of SDP, and DP, is mainly based on the unit propagation. In this subsection, I discuss the statistical behavior of the unit propagation when SDP is called with b bound variables and $(\le b)$ of them are bound by the unit propagation. Because SDP calls SDP recursively only in the splitting rule, there is no empty nor unit clause when b - 1 variables are bound. The number of variables whose values are assigned by the unit propagation, u, means u clauses selected for the unit propagation must be satisfied.

To analyze the statistical behavior of the unit propagation, I discuss three probabilities, the probability that the unit propagation binds at least Δu variables, $p_{unit}(b,u,\Delta u)$, the probability that SDP detects an empty clause after the unit propagation binds Δu variables, $p_{fail}(b,u,\Delta u)$, and the probability that the unit propagation terminates after binding Δu variables because there is no unit clause nor empty $p_{split}(b,u,\Delta u)$. Because of their definitions, they relate to each other through the following equations.

$$p_{unit}(b,u,0) = 1 \quad \cdots\cdots (1)$$
$$p_{unit}(b,u,\Delta u+1) = p_{unit}(b,u,\Delta u) \times \{1-p_{fail}(b,u,\Delta u)-p_{split}(b,u,\Delta u)\} \quad \cdots (2)$$

When $\Delta u=0$, i.e. when SDP is called, a clause that is not selected for the unit propagation before the current SDP is called, becomes unit with probability $p_{o\to u}(b)$ and does not become empty $p_{o\to e}(b-1,0,b,0)=0$ Thus,

$$p_{fail}(b,u,0) = 0 , \quad \cdots\cdots (3)$$
$$p_{split}(b,u,0) = \{1-p_{o\to u}(b)\}^{M-u} , \quad \cdots\cdots (4)$$

If $b \le K-2$, all unsatisfied clauses contain at least two literals and the unit propagation can not find an unit clause. SDP must apply the splitting rule (there must be an unbound variable be $N \ge K$) and the probabilities for the unit propagation are

$$p_{unit}(b,u,\Delta u) = \begin{cases} 1 & \Delta u = 0 \\ 0 & \Delta u \geq 1 \end{cases},$$

$$p_{fail}(b,u,0) = 0,$$

$$p_{split}(b,u,0) = 1.$$

I assume $b \geq K-1$ in the remaining part of this subsection.

When $\Delta u \geq 1$, p_{fail} and p_{split} depend on the number of unit clauses after the previous variable binding in the unit propagation. Let n be the number of unit clauses after the unit propagation binds $\Delta u - 1$ variables. The unit propagation selects one of n unit clauses and binds the variable in it to satisfy it. This value assignment may make a new empty clause from the remaining $n-1$ unit clauses. Because each unit clause becomes empty with probability $p_{u \to e}(b+\Delta u)$,

$$p_{fail}(b,u,\Delta u) = \sum_{n=1}^{M-u-\Delta u+1} p_{num}(b,u,\Delta u - 1, n) \times \left[1 - \left\{1 - p_{u \to e}(b+\Delta u)\right\}^{n-1}\right]. \quad \cdots(5)$$

Where, $p_{num}(b,u,\Delta u - 1, n)$ is the probability that there are n (≥ 1) unit clauses after the unit propagation binds $\Delta u - 1$ variables on the conditions that there is no empty clause and there is at least one unit clause, i.e. on the condition that the unit propagation binds one more variable.

The probability that there is no empty nor unit clause, $p_{split}(b,u,\Delta u)$, can be obtained as follows. To remove the remaining $n-1$ unit clauses, the variable binding for the selected unit clause must satisfy all of them and the binding must not make a new unit clause from $M-u-(\Delta u - 1)-n$ non-unit clauses (u clauses are already satisfied by the unit propagation before the current SDP is called, $\Delta u - 1$ are satisfied by the unit propagation in the current SDP and n are unit now). Thus, $p_{split}(b,u,\Delta u)$ is

$$p_{split}(b,u,\Delta u) = \sum_{n=1}^{M-u-\Delta u+1} p_{num}(b,u,\Delta u - 1, n) \times p_{u \to s}(b+\Delta u)^{n-1}\left\{1 - p_{o \to u}(b+\Delta u)\right\}^{M-u-\Delta u+1-n} \quad \cdots(6)$$

The remaining problem is to obtain the probability $p_{num}(b,u,\Delta u - 1, n)$. I calculate it as a recursion formula with respect to Δu. As I mentioned above, I need to calculate it on the conditions that no empty clause exists and at least one unit clause exists. Thus, a clause that is unit after the unit propagation binds $\Delta u - 1$ variables, is also unit after the Δuth binding with probability

$$p'_{u \to u}(b+\Delta u) = \frac{p_{u \to u}(b+\Delta u)}{1 - p_{u \to e}(b+\Delta u)}.$$

The probability that there are n' unit clauses after the Δuth binding in the unit propagation on the condition that there are n (≥ 1) unit clauses before the binding, is

$$p_t(b,u,\Delta u,n,n') = \sum_m \binom{n-1}{m} p'_{u \to u}(b+\Delta u)^m \left\{1 - p'_{u \to u}(b+\Delta u)\right\}^{n-1-m} \times \binom{M-u-\Delta u+1-n}{n'-m} p_{o \to u}(b+\Delta u)^{n'-m} \times \left\{1 - p_{o \to u}(b+\Delta u)\right\}^{M-u-\Delta u+1-n-(n'-m)}. \quad \cdots(7)$$

Where, m is the number of clauses that are unit before the Δuth variable binding in the unit propagation and also unit after the binding. This means that the variable binding causes $n'-m$ new unit clauses. The range of m is

$$\max(0, n' - (M-u-\Delta u+1-n)) \leq m \leq \min(n-1, n').$$

This probability leads the following recursion formula.

$$p_{num}(b,u,\Delta u,n') = \frac{\sum_{n=1}^{M-u-\Delta u+1} p_{num}(b,u,\Delta u - 1, n) p_t(b,u,\Delta u,n,n')}{1 - \sum_{n=1}^{M-u-\Delta u+1} p_{num}(b,u,\Delta u - 1, n) p_t(b,u,\Delta u,n,0)} \quad \cdots\cdots(8)$$

The denominator is a correction for the condition that there is at least one unit clause. When Δu=0, i.e. when the unit propagation starts, p_{num} is

$$p_{num}(b,u,0,n) = \frac{\binom{M-u}{n} p_{o \to u}(b)^n \left\{1 - p_{o \to u}(b)\right\}^{M-u-n}}{1 - \left\{1 - p_{o \to u}(b)\right\}^{M-u}} \quad \cdots(9)$$

because all of $M-u$ clauses, that are not selected for the unit propagation before the current SDP is called, are not empty nor unit when the splitting rule is applied and the value assignment to one variable by the splitting rule can not make an empty clause.

As a result, p_{num} can be obtained from (8)~(9) and with this probability, I can obtain the probabilities, p_{unit}, p_{fail} and p_{split} that decide the statistical behavior of the unit propagation from (1)~(6).

4.3 After the Unit Propagation

What happens after the unit propagation finishes? There are three possibilities, returning "unsatisfiable" because an empty clause exists, returning "satisfiable" because no empty clause exists and all variables are bound, and applying the splitting rule. The definitions of p_{unit}, p_{fail} and p_{split} lead the probabilities of these three cases as follows.

The unit propagation step binΔu variables and SDP

- returns "unsatisfiable"
  $p_{unit}(b,u,\Delta u)p_{fail}(b,u,\Delta u)$
- returns "sausfiable"
  $\begin{cases} 0 & \Delta u < N-b \\ p_{unit}(b,u,N-b)\{1-p_{fail}(b,u,N-b)\} & \Delta u = N-b \end{cases}$
- **applies the splitting rule**
  $\begin{cases} p_{unit}(b,u,\Delta u)p_{split}(b,u,\Delta u) & \Delta u < N-b \\ 0 & \Delta u = N-b \end{cases}$

4.4 Average Case Complexities

In this subsection, I discuss two average case complexities, the complexity for satisfiability checking and the complexity for finding all solutions. I use the number of SDP calls as the complexity.

Let $ts_{check}(b,u)$ and $ts_{all}(b,u)$ be the SDP's complexity for satisfiability checking and the complexity for finding all solutions when SDP is called with *b* bound variables and *u* of them are bound by the unit propagation. SDP calls SDP recursively only in the splitting rule and, for satisfiability checking, the splitting rule calls the second descendant SDP only when the first one returns "unsatisfiable". Thus,

$$ts_{check}(b,u) = 1 + \sum_{\Delta u=0}^{N-b-1} p_{unit}(b,u,\Delta u)p_{split}(b,u,\Delta u) \times \{2 - p_{SDP}(b+\Delta u+1, u+\Delta u)\} \times ts_{check}(b+\Delta u+1, u+\Delta u). \quad \cdots (10)$$

Where, $p_{SDP}(b,u)$ is the probability that SDP returns "sausfiable" when it is called with *b* bound variables and *u* of them are bound by the unit propagation. I discuss this probability in the next subsection. For finding all solutions, the splitting rule always calls the second descendent SDP.

$$ts_{all}(b,u) = 1 + 2\sum_{\Delta u=0}^{N-b-1} p_{unit}(b,u,\Delta u)p_{split}(b,u,\Delta u) \times ts_{all}(b+\Delta u+1, u+\Delta u) \quad \cdots\cdots (11)$$

These two recursion formulas reduce the complexities of SDP with *N-b* unbound variables to the complexities with less number of unbound variables. When *b=N,* i.e. when all variables are bound, SDP does not apply the splitting rule and

$$ts_{check}(N,u) = ts_{all}(N,u) = 1.$$

When SDP is called first, i.e. when no variable is bound, the unit propagation does not bind a variable because a ci$K$-SAT ($K \geq 2$) does not contain an unit clause. SDP applies the splitting rule and the complexity of each descendant SDP is $ts_{check}(1,0)$ or $ts_{all}(1,0)$. Thus the average case complexities for k-SAT are

$$ac_{check} = 1 + \{2 - p_{SDP}(1,0)\}ts_{check}(1,0), \quad \cdots\cdots\cdots\cdots (12)$$
$$ac_{all} = 1 + 2ts_{all}(1,0). \quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots (13)$$

4.5 Probability of Satisfiability

This subsection discuss the prot$p_{SDP}(b,u)$, that SDP returns "sausfiable" when it is called with *b* bound variables and *u* of them are bound by the unit propagation. The probability is, of course, a probability that the cnf with the partial value assignment to *b* variables is sausfiable.

Let $p_s(b_1,u_1,b_2,r_2)$ be the probability that the cnf with b_2 bound variables and r_2 removed variables is sausfiable on the conditions that the cnf does not contain an empty nor unit clause and contains at least u_1 satisfied clauses when b_1 variables are bound and no variable is removed. Because SDP applies the splitting rule and calls SDP recursively only when there is no empty nor unit clause, the probability of returning "sausfiable" is

$$p_{SDP}(b,u) = p_s(b-1,u,b,0). \quad \cdots\cdots\cdots\cdots\cdots\cdots (14)$$

When $b_2 + r_2 < N$, let *x* be a variable that is not bound nor removed. The cnf with b_2 bound variables and r_2 removed variables, is sausfiable iff one of the two cnfs, the cnf generated by assigning *true* to *x* and the cnf generated by assigning *false* to *x,* is sausfiable. Because of the definition of p_s, each of the two cnfs is sausfiable with probability $p_s(b_1,u_1,b_2+1,r_2)$. To obtain $p_s(b_1,u_1,b_2,r_2)$, I need the probability that both of the two cnfs are sausfiable. It is very difficult to calculate it and I approximate it with its lower bound, the probability that the two cnfs share an assignment that satisfies both of them. In other words, the lower bound is the probability that the conjunction of the two cnfs is sausfiable. The conjunction is generated by removing *x* from the cnf and the lower bound is $p_s(b_1,u_1,b_2,r_2+1)$. This is the reason I introduced 'removed variables" to the analysis. The above discussion leads the following recursion formula.

$$p_s(b_1,u_1,b_2,r_2) = \min(1, 2p_s(b_1,u_1,b_2+1,r_2) - p_s(b_1,u_1,b_2,r_2+1)) \quad (15)$$

When $b_2 + r_2 = N$, i.e. when all variables are bound or removed, a clause in the cnf is satisfied with probability $1 - p_{o\to e}(b_1,0,b_2,r_2)$ and

$$p_s(b_1,u_1,b_2,r_2) = \{1 - p_{o\to e}(b_1,0,b_2,r_2)\}^{M-u_1} \cdots (16)$$

The proba$p_{SDP}(b,u)$, can be obtained fron (14)~(16).

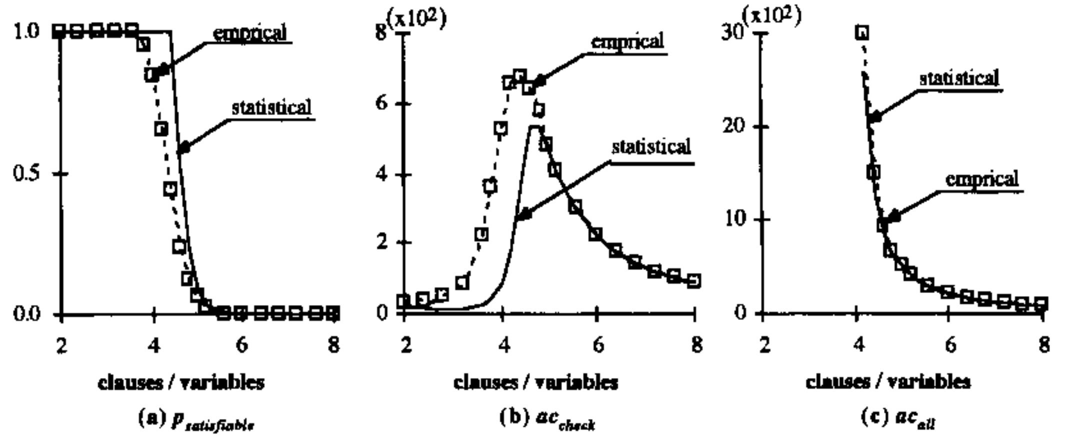

Figure 3: 3-SAT with 50 variables

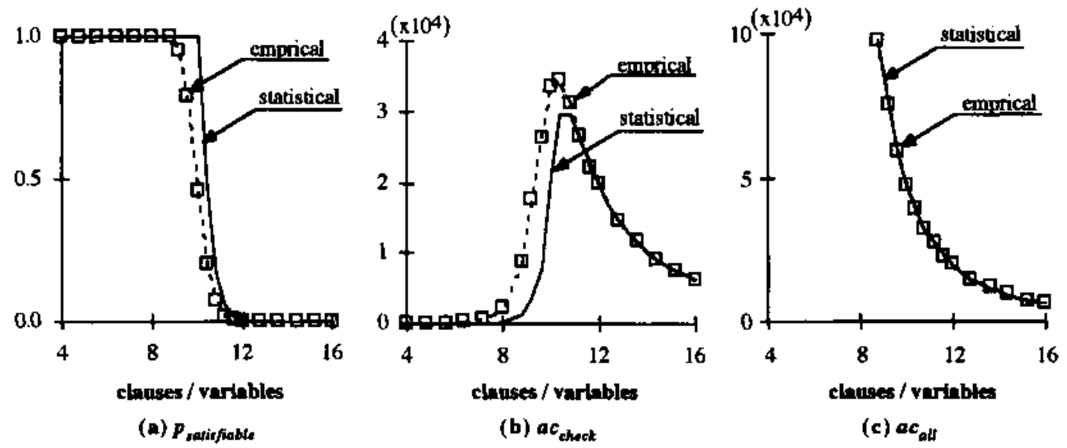

Figure 4: 4-SAT with 50 variables

Because a cnf generated on K-SAT ($K \geq 2$) contains neither empty nor unit clause, the probability of satisfiability for K-SAT is

$$p_{satisfiable} = p_t(0,0,0,0)\,. \cdots\cdots\cdots\cdots (17)$$

4.6 The complexity for computing complexities

It is not so easy to calculate the complexities with the result of this section because the result contains many recursion formulas. The bottleneck of the computation is to calculate $p_t(b,u,\Delta u,n,n')$. It requires $O(M)$ time to compute p_t for the given b, u, Δu, n, n' and $O(N^2M^3)$ to compute all elements of p_t because of the following relation.

$$p_t(b,u,\Delta u,n,n') = p_t(b-1,u-1,\Delta u+1,n,n')$$

It is polynomial time but not practical when N and M are large. But we don't need to calculate all elements of p_t because many of them are probabilities for very rare possibilities. For example, when b, u, Δu and n are fixed, the average of n' is

$$\begin{aligned} < n' > &= (n-1)p'_{u\to u}(b+\Delta u) \\ &+ (M-u-\Delta u+1-n)p_{o\to u}(b+\Delta u) \end{aligned}$$

and the standard deviation is

$$\begin{aligned} SD = \big[&(n-1)p'_{u\to u}(b+\Delta u)\{1-p'_{u\to u}(b+\Delta u)\} \\ &+ (M-u-\Delta u+1-n)p_{o\to u}(b+\Delta u)\{1-p_{o\to u}(b+\Delta u)\}\big]^{\frac{1}{2}}. \end{aligned}$$

The approximation, $p_t(b,u,\Delta u,n,n') = 0$, is reasonable when $|(n' - <n'>)/SD|$ is large.

Another reasonable approximation is ignoring the probabilities of the unit propagation for large Δu. Because $p_{unit}(b,u,\Delta u)$ monotonically decreases with respect to Δu, if $p_{unit}(b,u,\Delta u)$ becomes very small, the approximation

$$p_{unit}(b,u,\Delta u') = 0 \quad \Delta u' > \Delta u$$

does not affect the result. This approximation also decreases the number of elements of p_t to be calculated and shortens the computation time.

5 Comparison with Empirical Results

Figure 3 and 4 show the statistical and empirical results of the probability of satisfiability, $p_{satisfiable}$, the complexity for satisfiability checking, α_{check}, and the complexity for finding all solutions, α_{all}, for 3-SAT and 4-SAT with 50 variables. The complexities are the number of SDP calls that are required for satisfiability checking and for finding all solutions. The empirical results were obtained by solving randomly generated 1,000 problems at each clauses-to-variables ratio for 3-SAT and by solving 100 problems at each ratio for 4-SAT.

The probability of satisfiability coincided with the empirical result well. The 50% satisfiable points by the statistical analysis (4.6 for 3-SAT and 10.4 for 5-SAT) were a little larger than the empirical results (4.3 for 3-SAT and 10.0 for 4-SAT). This difference is caused by the approximation I used for calculating the recursion formula for p_s. The approximation makes p_3 larger than the real value and causes the overestimatiop_{SDP} and*Psatisfiable*-

The complexity for finding all solutions coincided with the empirical result very well at all clauses-to-variables ratios. On the other hand, the complexity for satisfiability checking coincided with me empirical result above the 50% satisfiable point, but became smaller than the experimental result bellow the point. It became maximum near the 50% satisfiable point and the statistical results at the peak (532 for 3-SAT and 29,500 for 4-SAT) were about 20% smaller for 3-SAT and about 15% smaller for 4-SAT than the empirical results (671 for 3-SAT and 34,700 for 4-SAT). These results suggests that the analysis of the unit propagation, that are commonly used for both of two complexities, gives the accurate result. The overestimauon of p_{SDP} decreases the probability of calling the second descendant SDP and decreases the complexity for satisfiability checking. Above the 50% satisfiable point, the statistical prediction for satisfiability checking coincided with the empirical result because p_{SDP} becomes very small and the approximation for it does not affect the statistical result.

6 Conclusions

In this paper, I analyzed the average case behavior of the Davis-Putnam procedure on the fixed clause length model. The analysis included two average case complexities, the complexity for satisfiability checking and the complexity for finding all solutions. I also discussed the probability of satisfiability.

The results, especially the complexity for finding all solutions, coincided with experimental results well. The complexity for satisfiability checking coincided with experiments above the 50% satisfiable point but became smaller below the point because of the overestimation of the probability that a cnf with a partial assignment is satisfiable.

References

[Chao and Franco, 90] Chao, M. and Franco, J., Probabilistic Analysis of a Generalization of the Unit-Clause Literal Selection Heuristics for the k-Satisfiability Problem. *Information Science,* Vol. 51 (1990), 289-315.

[Crawford and Auton, 93] Crawford, J.M. andAuton, L.D., Experimental Results on the Crossover Point in Satisfiability Problems. *Proc. of AAAI-93* (1993), 21-27.

[Davis and Putnam, 60] Davis, M. and Putnam, H., A Computing Procedure for Quantification Theory. *J. Assoc. Comput. Mack,* Vol. 7 (1960), 201-215.

[Franco, 86] Franco, J., On the Probabilistic Performance of Algorithms for the Satisfiability Problem. *Information Processing Letters,* Vol. 23 (1986), 103-106

[Franco and Paull, 83] Franco, J. and Paull, M., Probabilistic Analysis of the Davis-Putnam Procedure for Solving the Satisfiability Problems. *Discrete Applied Mathematics,* Vol. 5 (1983), 77-87.

[Franco, Plotkin and Rosenthal, 87] Franco, J., Plotlrin, J.M. and Rosenthal, J.W., Correction to Probabilistic Analysis of the Davis-Putnam Procedure for Solving the Satisfiability Problems. *Discrete Applied Mathematics,* Vol. 17 (1987), 295-299.

[Goldberg, Purdom and Brown, 82] Goldberg, A., Purdom, P. and Brown, C, Average Time Analysis of Simplified Davis-Putnam Procedures. *Information Processing Letters,* Vol. 15 (1982), 72-75.

[Mitchell, Selman and Levesque, 92] Mitchell, D., Selman, B. and Levesque, H., Hard and Easy Distributions of SAT Problems. *Proc. of AAAI-92* (1992), 459,465.

[Selman, Levesque and Mitchell, 92] Selman, B., Levesque, H. and Mitchell, D., A New Method for Solving Hard Satisfiability Problems. *Proc. of AAAI-92* (1992), 440-446.

[Williams and Hogg, 94] Williams, C.P., and Hogg, T., Exploiting the deep structure of constraint problems. *Artificial Intelligence,* Vol. 70 (1994), 73-117.

[Yugami, Ohta and Hara, 94] Yugami, N., Ohta, Y. and Hara, H., Improving Repair-based Constraint Satisfaction Methods by Value Propagation. *Proc. of AAAI-94* (1994), 344-349.