Master Thesis

Investigating and improving the PPSZ algorithm for SAT

Author[s]:
Hertli, Timon

Publication Date:
2010

Permanent Link:
https://doi.org/10.3929/ethz-a-006206989

Rights / License:
In Copyright - Non-Commercial Use Permitted

This page was generated automatically upon download from the ETH Zurich Research Collection. For more information please consult the Terms of use.
Investigating and Improving the PPSZ Algorithm for SAT

Master Thesis
Timon Hertli
October 2010

Advisors: Prof. Dr. Emo Welzl, Robin Moser, Dominik Scheder
Department of Computer Science, ETH Zürich
Abstract

In this thesis, we give a self-contained analysis of the PPSZ algorithm [8], including the combination with Schöning’s algorithm [14] of Iwama and Tamaki [5] and the improvement of Rolf [13].

We also give new bounds for 3-SAT and 4-SAT using the following idea: A critical variable of a satisfiable CNF formula is a variable that has the same value in all satisfying assignments. With a simple case distinction on the fraction of critical variables of a CNF formula, we improve the bound for 3-SAT from $O(1.32216^n)$ [13] to $O(1.32153^n)$. Using a different approach, Iwama et al. [4] very recently achieved a running time of $O(1.32113^n)$. Our method nicely combines with theirs, yielding an even faster algorithm with running time $O(1.32065^n)$. We also improve the bound for 4-SAT from $O(1.47390^n)$ [5] to $O(1.46928^n)$, where $O(1.46981^n)$ can be obtained using only the methods of [5] and [13]. This is very close to the bound for unique 4-SAT for PPSZ, $O(1.46899^n)$ [8].
# Contents

## 1 Introduction

## 2 Notation and Basic Concepts
  2.1 Notation
  2.2 Basic Concepts of SAT

## 3 General Properties and Unique SAT
  3.1 The Algorithms
  3.2 Basic Properties
  3.3 Basic Concepts of The Analysis
  3.4 Critical Clause Trees
  3.5 Analysis for Unique 3-SAT
  3.6 Analysis for Unique k-SAT

## 4 General k-SAT
  4.1 Subcubes and Critical Clause Trees
  4.2 Nice Distribution Functions
  4.3 Proof of (4.3) and (4.4)
  4.4 Bound for k\(\geq 5\)
  4.5 Schöning’s Algorithm
  4.6 Optimized Distribution Functions for 3-SAT

## 5 Improved Analysis for 3-SAT and 4-SAT
  5.1 Critical Variables
  5.2 A Better Bound for PPSZ
  5.3 3-SAT
    5.3.1 Our Improvement
    5.3.2 Combination with Improvement of [4]
## Contents

5.4 4-SAT ................................................. 60

6 Conclusion ........................................... 61

Bibliography ........................................... 63
Chapter 1

Introduction

The boolean satisfiability problem (SAT) for logical formulas in conjunctive normal form (CNF) is an important and well-studied problem. In this thesis, we consider $k$-SAT, where every clause has size at most $k$. We especially consider the cases $k = 3$ and $k = 4$.

Let $n$ denote the number of variables of a CNF formula. A trivial $k$-SAT algorithm runs in time $O(poly(n) \cdot 2^n)$ by trying all assignments. A lot of research has been done to obtain faster algorithms. If $P \neq NP$ (which is widely believed), there exists no polynomial time algorithm for $k$-SAT. The algorithms known so far have “moderatly” exponential running time, i.e. $O(b^n)$ (for $b < 2$) or equivalently $O(2^{cn})$ (for $c < 1$). The fastest algorithms known today are randomized algorithms that try to find a satisfying assignment with some probability, provided one exists. Repeating such a try a number of times inverse to the success probability then gives an exponential time algorithm that decides satisfiability with high probability.

The PPSZ algorithm [8] that is the focus of this thesis has its roots in the PPZ algorithm. In 1999, Paturi, Pudlák, and Zane [9] proposed the following algorithm: Given a $k$-CNF formula $F$, we want to find a satisfying assignment of $F$. The $n$ variables are processed step by step according to a random permutation. At each step a variable $x$ is substituted by a value chosen as follows: If there is exactly one corresponding unit clause ($\{x\}$ or $\{\overline{x}\}$), $x$ is set such that this clause is satisfied. In this case, we call $x$ forced. If there is no corresponding unit clause, $x$ is chosen uniformly at random and we call $x$ guessed. If both unit clauses or the empty clause occur, we cannot find a satisfying assignment anymore and we abort.

Intuitively, if $F$ is “strongly constrained”, then the algorithm encounters many unit clauses; hence it needs to guess significantly fewer than $n$ variables. On the other hand, if $F$ is only “weakly constrained”, it has multiple
1. Introduction

satisfying assignments, making it easier to find one. Paturi, Pudlák and Zane [9] made this intuition precise and show that PPZ finds a satisfying assignment for a $k$-CNF formula with probability at least $2^{-\left(1-1/k\right)n}$, provided there exists one.

Paturi, Pudlák, Saks, and Zane [8] extended this algorithm to the PPSZ algorithm by combining PPZ with bounded resolution, a restricted form resolution where only clauses up to some size are considered. The PPSZ algorithm [8] is the currently fastest known algorithm for unique $k$-SAT, where $F$ has either exactly one or no satisfying assignment. For this case, PPSZ has even been derandomized by Rolf [12]. For unique 3-SAT, the running time is $O(1.30704^n)$. For $k \geq 5$, it was shown in [8] that the unique case is the worst case and so the same bound applies also to general $k$-SAT. For 4-SAT and especially for 3-SAT, the analysis results in a gap between the unique and the general case. In addition, the analysis in the general case is very complicated. For 3-SAT, the analysis of [8] gives a running time of $O(1.3633^n)$.

In 1999, Schöning [14] came up with a very simple random walk algorithm. In Schöning’s algorithm, first a random assignment is chosen. Then for at most $3n$ times, a dissatisfied clause $C$ is picked and the value of a variable chosen u.a.r. from $C$ is flipped. This algorithm achieves running time $O(1.3334^n)$ for 3-SAT. Schöning’s algorithm has very recently been derandomized by Moser and Scheder [6], giving the fastest known deterministic algorithm for 3-SAT. Schöning’s algorithm has been repeatedly improved in [3, 10, 1, 11] to $O(1.3302^n)$, $O(1.32971^n)$, $O(1.3290^n)$, and $O(1.32793^n)$.

In 2004, Iwama and Tamaki [5] observed that PPSZ can be combined with Schöning’s algorithm to obtain a better combined bound. Intuitively, Schöning’s algorithm performs better when there are many satisfying assignments, and PPSZ performs better when there are few satisfying assignments. In [5], this was made formal, improving the bound to $O(1.32373^n)$. In 2006, Rolf [13] improved this to $O(1.32216^n)$ by giving a more refined analysis.

In this thesis, we further improve the bound for general 3-SAT and 4-SAT. For 3-SAT, we improve the result of [13] to obtain the bound $O(1.32153^n)$. Very recently, Iwama et al. [4] showed how to adapt the idea of [3] and [1] to improve Schöning in the combined algorithm, achieving a running time of $O(1.32113^n)$. Our approach nicely combines with theirs and we achieve a combined bound $O(1.32065^n)$. For 4-SAT, we obtain $O(1.46928^n)$, where $O(1.46981^n)$ can be obtained using the methods of [5] and [13]. Our bound for 4-SAT is very close to the bound for unique 4-SAT, $O(1.46899^n)$ [8].

The PPSZ papers. There are two versions of [8], which we call the old version and the new version. For unique k-SAT, both are the same. For
general k-SAT, the old version of [8] gives a more complicated analysis. The old version gives a better bound for 3-SAT and the new version gives a better bound for 4-SAT.

Only the new version is published, but the old version is still available at the Citeseer cache\footnote{http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.1134}. We have found some errors in that version, but they are not so hard to correct. There is also a conference version [7] stating the results of the old version of [8], but without most proofs.

In Iwama and Tamaki [5], the new version of [8] is referenced, but for the calculations the old version is used. Just using the new version would give even better bounds for 3-SAT, as stated by Rolf [13]. However [13] does not consider 4-SAT. We improve 4-SAT by using the ideas of [13] together with our own improvement.

**Improvement using Critical Variables.** A critical variable of a satisfiable CNF $F$ is a variable that has the same value in all satisfying assignments. Denote by $c(F)$ the fraction of variables that are critical for $F$.

Our improvement is now based on two observations:

1. Assigning a random value to a random variable preserves satisfiability with probability $1 - \frac{c(F)}{2}$.

2. In PPSZ, critical variables behave in some sense as in the unique case and are thus forced with a larger probability.

Assume there is a randomized algorithm $A$ that finds a satisfying assignment of a satisfiable $k$-CNF $F$ with probability $\left(1 - \frac{c^*}{2}\right)^n$ given $c(F) \geq c^*$ for some $c^* \in [0, 1]$. Denote by Guess-$A$ the algorithm that repeats the following $n$ times: We call $A$ on $F$; if the returned assignment does not satisfy $F$, we choose a variable u.a.r. and replace it in $F$ by a value chosen u.a.r. from $\{0, 1\}$.

It is not hard to see that Guess-$A$ finds a satisfying assignment with probability at least $\left(1 - \frac{c^*}{2}\right)^n$ for all $k$-CNFs $F$. For the correct choice of $c^*$, a better bound of PPSZ given $c(F) \geq c^*$ implies now a better bound of Guess-PPSZ.

In this thesis, we first provide a self-contained analysis of the PPSZ algorithm and the combination with Schöning’s algorithm, combining the results of [8, 14, 5, 13]. In Chapter 3, we give the analysis for unique $k$-SAT of [8]. In Chapter 4, we show how to extend the analysis to general $k$-SAT and we show the result of [13] for 3-SAT. In Chapter 5, we present our improvement and prove the mentioned new bounds for 3-SAT and 4-SAT.
Acknowledgements

I am very grateful to my advisors Robin Moser, Dominik Scheder and Emo Welzl for their many helpful suggestions and great support.
Chapter 2

Notation and Basic Concepts

In this section we introduce the notation of the thesis and introduce basic concepts of SAT. We use the notational framework introduced by Welzl [16] with some modifications.

2.1 Notation

General. By $[n]$ we denote the set of integers from 1 to $n$: $[n] := \{1, 2, \ldots, n\}$. By $\text{poly}(n)$ we denote some arbitrary but fixed polynomial in $n$. This is used to denote polynomial factors in the running time of exponential time algorithms.

Let $\mathbb{N} := \{1, 2, 3, \ldots\}$, the set of natural numbers without 0. Let $\mathbb{N}_0 := \{0, 1, 2, \ldots\}$, the set of natural numbers with 0.

Functions. We consider a function as a set of pairs $x \mapsto y$. For a function $f : X \to Y$ and a set $S$, we denote by $f|_S$ the function $g$ from $X \cap S \to Y$ s.t. for all $t \in X \cap S$ we have $g(t) = f(t)$.

Let $f : X \to Y$. For some set $S$ with $S \not\subseteq X$ (to prevent ambiguity), we denote by $f(S) := \{y \in Y : \exists x \in S \text{ with } y = f(x)\}$, the image of $S$ under $f$.

Randomness. By choosing an element u.a.r. from a finite set, we mean choosing it uniformly at random from all elements of that set. By choosing an element u.a.r. from a closed real interval, we mean choosing it according to the continuous uniform distribution over this interval. Unless otherwise stated, all random choices are mutually independent.
2. Notation and Basic Concepts

A (uniformly) random permutation \( \pi \) on some finite set \( V \) is a permutation chosen u.a.r. from the set of all permutations on \( V \). We view a permutation on \( V \) as an ordering of the elements of \( V \).

**CNF and SAT.** For us a CNF formula is a finite set of clauses, where a clause is a finite set of literals, and where a literal is a variable \( x \) (a positive literal) or its negation denoted by \( \overline{x} \) (a negative literal). We usually just write CNF instead of CNF formula. The literal \( x \) or \( \overline{x} \) is said to be over the variable \( x \). We implicitly treat a variable as a positive literal. We assume that no clause contain both a variable and its negation. The negation \( \overline{l} \) of a negative literal \( l \) is the variable the literal is over. For a set of literals \( L \), we denote by \( L' \) the set of literals \( l \) where \( l \in L \).

A CNF formula is interpreted as a logical formula in conjunctive normal form by joining the literals in a clause by a logical OR, and then joining the clauses by a logical AND. For example the CNF \( \{\{x, \overline{y}\}, \{z\}\} \) is interpreted as \((x \lor \overline{y}) \land (z)\).

We denote a clause that has exactly \( k \), at least \( k \) or at most \( k \) literals as a \( k \)-clause, \((\geq k)\)-clause or \((\leq k)\)-clause, respectively. We denote a CNF that contains only \( k \)-clauses, \((\geq k)\)-clauses or \((\leq k)\)-CNF, respectively. We sometimes write unit clause instead of 1-clause.

Let \( C \) be a clause. By \( V(C) \), we denote the set of variables of \( C \). Let \( F \) be a CNF. By \( V(F) \), we denote the set of variables of \( F \). By \( n(F) := |V(F)| \), we denote the number of variables of \( F \). An assignment \( \alpha \) on \( V \) is a function mapping \( V \to \{0,1\} \). We say that an assignment \( \alpha \) on \( V \supseteq V(F) \) satisfies \( F \) if the logical formula evaluates to true when each variable \( x \in V(F) \) is replaced by \( \alpha(x) \). By \( \text{sat}(F) \) we denote the set of satisfying assignments on \( V(F) \) of \( F \). If \( \text{sat}(F) \neq \emptyset \), we say that a \( F \) is satisfiable. The all-one assignment is the assignment that assigns the value 1 to any variable.

A partial assignment on \( V \) is a function mapping some subset \( W \) of \( V \) to \( \{0,1\} \). For a CNF \( F \) and some partial assignment \( \alpha \) on \( V(F) \), let \( F[\alpha] \) be \( F \) restricted to \( \alpha \), i.e. the CNF obtained by replacing variables that are set in \( \alpha \) with their values in the logical formula. We also write \( F[\{x \mapsto a\}] \) instead of \( F[\{x \mapsto a\}] \).

Let \( \alpha \) be an assignment on \( V \), and let \( S \subseteq V \). By \( \alpha \oplus S \), we define the assignment obtained from \( \alpha \) by flipping the variables of \( S \) (i.e. replacing 0 by 1 and vice versa).

The (Hamming) distance between two assignments \( \alpha \) and \( \beta \) on the same set \( V \) is the number of variables in which \( \alpha(x) \neq \beta(x) \). Let \( F \) be a satisfiable CNF, \( \alpha \in \text{sat}(F) \) and \( d \in \mathbb{N}_0 \). We say that \( \alpha \) is \( d \)-isolated, if any assignment \( \beta \) with distance at most \( d \) to \( \alpha \) does not satisfy \( F \).
2.2. Basic Concepts of SAT

**Definition 2.1** Let $F$ be a satisfiable CNF and $\alpha \in \text{sat}(F)$. For $x \in V(F)$, we say that $\alpha$ is $d$-isolated given flipped $x$, if any assignment $\beta$ with $\beta(x) \neq \alpha(x)$ and distance at most $d$ to $\alpha$ does not satisfy $F$.

**Logarithms.** We denote by log the logarithm to the base 2. For the logarithm to the base $e$, we write $\ln$. We define $0 \log 0 := 0$.

2.2 Basic Concepts of SAT

**Resolution.** We say that two clauses $C_1$ and $C_2$ conflict in a variable $x$, if one of them contains $x$ and the other contains $\overline{x}$. We call $(C_1, C_2)$ a resolvable pair, if they conflict in exactly one variable $x$, and we define their resolvent by $R(C_1, C_2) := (C_1 \cup C_2) \setminus \{x, \overline{x}\}$. It is easy to see that if $F$ contains a resolvable pair $(C_1, C_2)$, then $\text{sat}(F) = \text{sat}(F \cup R(C_1, C_2))$.

A resolvable pair $(C_1, C_2)$ is $s$-bounded, if $|C_1| \leq s$, $|C_2| \leq s$, and $|R(C_1, C_2)| \leq s$.

We call a sequence of clauses $(C_1, C_2, \ldots, C_n)$ a resolution deduction of $C_n$ from a CNF $F$, if the following holds: For all $i \in \{1, \ldots, n\}$, either $C_i \in F$ or there is a resolvable pair $(C_j, C_k)$ with $j < i$ and $k < i$, s.t. $C_i = R(C_j, C_k)$. We call a resolution deduction $s$-bounded, if for all $i \in \{1, \ldots, n\}$, $|C_i| \leq s$.

By $\text{RESOLVE}(F, s)$, we denote the CNF obtained by adding to $F$ all clauses that have an $s$-bounded resolution deduction from $F$. By a straightforward algorithm, we can compute $\text{RESOLVE}(F, s)$ in time $O(n(F)^3 \text{poly}(n(F)))$ [8].

**Stronger and Weaker CNFs.** We now define a concept that generalizes the subset relation of CNFs:

**Definition 2.2** Let $F$ and $G$ be CNFs. If for every clause $C \in F$ there is a clause $C' \subseteq C$ in $G$, we say that $F$ is weaker than $G$ and $G$ is stronger than $F$. We denote this by $F \leq G$ and $G \geq F$.

It is easy to check that these relations are partial orders. Note that $F \subseteq G$ implies $F \leq G$, and $F \leq G$ implies $\text{sat}(F) \supseteq \text{sat}(G)$ (if the assignments are considered on the same variable set). The stronger relation is preserved by (s-bounded) resolution:

**Lemma 2.3** Let $F$ and $G$ be CNFs with $F \leq G$. If there is an (s-bounded) resolution deduction of some clause $C$ from $F$, then there exists a clause $D \subseteq C$ s.t. there is an (s-bounded) resolution deduction of $D$ from $G$.

This implies $\text{RESOLVE}(F, s) \leq \text{RESOLVE}(G, s)$.
Proof Let \((C_1, \ldots, C_n = C)\) be an \((s\text{-bounded})\) resolution deduction from \(F\) used to obtain \(C\). We construct a resolution deduction \((D_1, \ldots, D_n = D)\) from \(G\) with the property that \(D_i \subseteq C_i\). Hence if \((C_1, \ldots, C_n)\) is \(s\text{-bounded},\) then \((D_1, \ldots, D_n)\) is also \(s\text{-bounded}.

We do induction on \(i\) as follows: Let \(i \in [n]\) and assume that we have a resolution deduction \((D_1, \ldots, D_{i-1})\) with \(D_j \subseteq C_j\) for \(j \in [i-1]\). We want to construct \(D_i \subseteq C_i\) s.t. we can add \(D_i\) to the resolution deduction. If \(C_i \in F\), we let \(D_i\) be a clause in \(G\) with \(D_i \subseteq C_i\), which exists as \(G\) is stronger than \(F\). Otherwise, we show that if \(R(C_j, C_k) = C_i\), then \((D_j, D_k)\) is a resolvable pair with \(R(D_j, D_k) \subseteq C_i\), or \(D_j \subseteq C_i\) or \(D_k \subseteq C_i\). With this, the existence of \(D_i\) is obvious. Note that we can always repeat clauses in a resolution deduction. If \((D_j, D_k)\) is a resolvable pair, then \(R(D_j, D_k) \subseteq C_i\) is trivial. Assume that \(C_j\) and \(C_k\) conflict in \(x\). If \((D_j, D_k)\) is no resolvable pair, this can only be because \(D_j\) and \(D_k\) do not conflict, and one clause must contain no literal over \(x\). If this is \(D_j\), and so \(D_j \subseteq C_j\) \(\setminus \{x, \overline{x}\} \subseteq C_i\). The case for \(D_k\) follows by symmetry.

Doing bounded resolution after restricting a formula gives a stronger formula than doing it before restricting:

Lemma 2.4 For a CNF \(F\), \(x \in V(F)\), \(a(x) \in \{0, 1\}\) and \(s \in \mathbb{N}_0\), we have

\[
\text{RESOLVE}(F, s)^{[x \rightarrow a(x)]} \leq \text{RESOLVE}(F^{[x \rightarrow a(x)]}, s).
\]

As the stronger-relation is is transitive, this also applies to restricting to partial assignments that set more than one variable.

Proof We assume w.l.o.g. that \(a(x) = 1\). Let \(C' \in \text{RESOLVE}(F, s)^{[x \rightarrow 1]}\). We have to show that there exists a clause \(D \in \text{RESOLVE}(F^{[x \rightarrow a(1)]}, s)\) with \(D \subseteq C'\). We have \(C'\) or \(C' \cup \{x\}\) in \(\text{RESOLVE}(F, s)\); let such a clause be denoted by \(C\). Note that \(x \notin C\). If \(C \in F\), the statement is trivial. Otherwise, let \((C_1, \ldots, C_n = C)\) be an \(s\text{-bounded} resolution deduction of \(C\) from \(F\). We want to construct an \(s\text{-bounded} resolution deduction of a clause \(D \subseteq C\) from \(F^{[a \rightarrow 1]}\). As \(\{x\} \notin D\), this implies \(D \subseteq C'\).

Let \(G := F^{[x \rightarrow 1]} \cup \{\{x\}\}\); it is easy to check that \(F \leq G\). Hence there is a clause \(D \in \text{RESOLVE}(G, s)\) with \(D \subseteq C\) by Lemma 2.3. Consider an \(s\text{-bounded} resolution deduction \((D_1, \ldots, D_n = D)\) of \(D\) from \(G\). As \(D \subseteq C\), we have \(x \notin D\). Also \(\{x\}\) does not form a resolvable pair with any clause in \(G\). Therefore we can remove all occurrences of \(\{x\}\) in the resolution derivation to obtain an \(s\text{-bounded} resolution derivation of \(D\) from \(F^{[x \rightarrow 1]}\). □
General Properties and Unique SAT

In this chapter, we introduce the PPZ [9] and PPSZ [8] algorithms and provide the analysis from [8] if there is a unique satisfying assignment. In the next chapter, we consider general satisfiability. In this thesis, let \( k \) be a fixed integer at least 3.

3.1 The Algorithms

We define the following algorithms in pseudocode from [8]:

\[
\text{MODIFY}(F, \pi, \beta), \quad \text{PPZ}(F), \quad \text{PPZ}(F, \beta), \quad \text{PPSZ}(F, s), \quad \text{PPSZ}(F, s, \beta),
\]

where \( F \) is a CNF, \( \pi \) is a permutation on \( V(F) \), \( \beta \) is an assignment on \( V(F) \), and \( s \in \mathbb{N}_0 \). This arrangement from [8] makes it clear that PPSZ is just PPZ with bounded resolution. If possible, we will define properties first for PPZ and then we show how to extend them to PPSZ. This requires some technical but simple lemmas provided in the previous chapter.

\text{MODIFY}(F, \pi, \beta) \) is the core algorithm, the others can be seen as wrappers adding random choices and bounded resolution. We call the steps in the loop of \text{MODIFY} also PPZ-steps or PPSZ-steps. We make the following definitions concerning the probabilities that PPZ and PPSZ return a fixed assignment or an arbitrary satisfying assignment.

**Definition 3.1** Let

\[
p_{\text{PPZ}}(\alpha, F) := \Pr(\text{PPZ}(F) = \alpha)
\]

and

\[
p_{\text{PPSZ}}(\alpha, F, s) := \Pr(\text{PPSZ}(F, s) = \alpha).
\]

The following probabilities are called success probabilities. Let

\[
p_{\text{PPZ}}(F) := \Pr(\text{PPZ}(F, s) \in \text{sat}(F)) = \sum_{\alpha \in \text{sat}(F)} p_{\text{PPZ}}(\alpha, F),
\]
3. General Properties and Unique SAT

**Algorithm 1** \texttt{Modify}(CNF $F$, permutation $\pi$, assignment $\beta$)

Let $\alpha$ be a partial assignment on $V(F)$, initially the empty assignment.

$i \leftarrow 0$

$F_0 \leftarrow F$

\textbf{for} all $x \in V(F)$, according to $\pi$ \textbf{do}

\begin{itemize}
  \item $i \leftarrow i + 1$
  \item \{we are now in step $i$, the PPZ-step of the variable $x$\}
  \item \{check if there is a 1-clause of $x$\}
  \item \textbf{if} $\{x\} \in F_{i-1}$ \textbf{then}
    \begin{itemize}
      \item $\alpha(x) \leftarrow 1$
    \end{itemize}
  \item \textbf{else if} $\{\overline{x}\} \in F_{i-1}$ \textbf{then}
    \begin{itemize}
      \item $\alpha(x) \leftarrow 0$
    \end{itemize}
  \item \textbf{else}
    \begin{itemize}
      \item \{otherwise choose the value of $x$ according to $\beta$\}
      \item $\alpha(x) \leftarrow \beta(x)$
    \end{itemize}
  \end{itemize}

end if

$F_i \leftarrow F\left[x \mapsto \alpha(x)\right]_{i-1}$

\textbf{end for}

\textbf{return} $\alpha$

**Algorithm 2** \texttt{PPZ}(CNF $F$)

For all $x \in V(F)$, choose $\beta(x)$ u.a.r. from $\{0, 1\}$

\textbf{return} \texttt{PPZ($F, \beta$)}

**Algorithm 3** \texttt{PPZ}(CNF $F$, assignment $\beta$)

Let $\pi$ be a random permutation on $V(F)$.

$\alpha \leftarrow \texttt{Modify}(F, \pi, \beta)$

\textbf{return} $\alpha$

**Algorithm 4** \texttt{PPSZ}(CNF $F$, integer $s$)

For all $x \in V(F)$, choose $\beta(x)$ u.a.r. from $\{0, 1\}$

\textbf{return} \texttt{PPSZ($F, s, \beta$)}

**Algorithm 5** \texttt{PPSZ}(CNF $F$, integer $s$, assignment $\beta$)

$F_s \leftarrow \texttt{Resolve}(F, s)$

\textbf{return} \texttt{PPZ($F_s, \beta$)}
the probability that PPSZ finds a satisfying assignment of $F$. Let

$$p_{PPSZ}(F, s) := \Pr(PPSZ(F, s) \in \text{sat}(F)) = \sum_{\alpha \in \text{sat}(F)} p_{PPSZ}(\alpha, F, s),$$

the probability that PPSZ finds a satisfying assignment of $F$ with parameter $s$.

As $PPSZ(F, s) = PPZ(\text{Resolve}(F, s))$ and $\text{sat}(\text{Resolve}(F, s)) = \text{sat}(F)$, we have $p_{PPSZ}(F, s) = p_{PPZ}(\text{Resolve}(F, s))$. This means that the success probability of PPSZ is just the success probability of PPZ on a preprocessed formula.

### 3.2 Basic Properties

We can characterize PPZ recursively, by first assigning a value u.a.r. or according to a unit clause, and then calling PPZ on the restricted formula. We will need this observation to show in the next chapter that if an assignment is “nearly” unique, we are essentially in the unique case.

**Observation 3.2** Let $F$ be a satisfiable CNF with $n(F) \geq 1$. Choose $x \in V(F)$ u.a.r. For all $y \in V(F)$, choose $\gamma(y)$ as follows: If there is a unit clause over $y$, choose $\gamma(y)$ accordingly. Otherwise choose $\gamma(y) \in \{0, 1\}$. Then for every assignment $\beta$ on $V(F)$, we have

$$\Pr(PPZ(F) = \beta) = \Pr(PPZ(F^{[x \rightarrow \gamma(x)]} \cup \{x \mapsto \gamma(x)\} = \beta).$$

For CNFs with the same set of satisfying assignments, $p_{PPZ}(F)$ is monotonically increasing in $F$ w.r.t. the stronger-relationship (see Definition 2.2):

**Lemma 3.3** Let $F$ and $G$ be CNFs. If $\text{sat}(F) = \text{sat}(G)$ and $F \geq G$, then $p_{PPZ}(F) \geq p_{PPZ}(G)$.

From Lemma 2.3 follows that if $F \geq G$, then $\text{Resolve}(F, s) \geq \text{Resolve}(G, s)$; hence we have also $p_{PPSZ}(F) \geq p_{PPSZ}(G)$.

**Proof** We show for fixed $\pi$ and $\beta$ that $\text{MODIFY}(F, \pi, \beta) \neq \text{MODIFY}(G, \pi, \beta)$ implies $\text{MODIFY}(G, \pi, \beta) \not\in \text{sat}(G)$. This directly implies the statement.

Assume $\text{MODIFY}(F, \pi, \beta) \neq \text{MODIFY}(G, \pi, \beta)$. Let $x$ be the first variable (w.r.t. $\pi$) where the returned assignments differ; this implies that $\{x\}$ or $\{\overline{x}\}$ occurs in at least one execution. Let $\alpha'$ be the assignment produced just before the step of $x$, which is the same in both executions. If both unit clauses occur, then $\alpha'$ cannot be extended to a satisfying assignment and hence $\text{MODIFY}(G, \pi, \beta) \not\in \text{sat}(G)$. 

11
If exactly one unit clause over $x$ is in $G^{[x]}$, then it is easy to see that this clause or the empty clause must be in $F^{[x]}$. In the former case, the returned assignments do not differ in $x$. In the latter case $F^{[x]}$ and hence $G^{[x]}$ is not satisfiable, implying \(\text{MODIFY}(G, \pi, \beta) \notin \text{sat}(G)\).

Otherwise we have either $\{\overline{x}\}$ or $\{x\}$ in $F$. $\beta$ does not satisfy this clause, as otherwise the returned assignments would not differ in $x$. Therefore $F^{[x] \cup \{x \mapsto \beta(x)\}}$ is not satisfiable, and as before we have $\text{MODIFY}(G, \pi, \beta) \notin \text{sat}(G)$.

In the analysis, we do bounded resolution to the size of some slowly growing function $s(n)$ in the number of variables. Let $s(n) : \mathbb{N} \to \mathbb{N}_0$ s.t. $s(n) \leq \log n$ and $\lim_{n \to \infty} s(n) = \infty$. For example, we could use $s(n) = \lceil \log \log(n) \rceil$ (with $s(1) := 0$). It is clear that $s(n)$-bounded resolution on a $(\leq k)$-CNF with $n$ variables takes time $2^{o(n)}$, which is negligible for our purposes. We do not require an upper bound as strong as $\log n$ for that. The limiting factor is not the time needed to do resolution but Lemma 4.13, where we require $s(n)^2 = o(\sqrt{n})$. Hence the bound $s(n) \leq \sqrt[5]{n}$ would also work, but $s(n) \leq \sqrt{n}$ would not.

### 3.3 Basic Concepts of The Analysis

In this section we will introduce the basic concepts from [8] for the analysis of PPSZ. In the following, let $F$ be a fixed satisfiable CNF.

We first introduce the concept of a placement, which is central for our analysis. A placement can be seen as an extended permutation, with the important property that variables can be considered independently.

**Definition 3.4 ([8])** A placement on the variable set $V$ is a function that assigns to each $x \in V$ a value from $[0, 1]$. The uniform random placement $\pi_U$ on $V$ is a random function that assigns to each $x \in V$ a value chosen u.a.r. from $[0, 1]$. A nice random placement on $V$ is a random function that assigns to each $x \in V$ a value independently from $[0, 1]$ (with a distribution depending on $x$).

When considering a CNF $F$, we write $\pi_U$ for the uniform random placement on $V(F)$. Ordering a set according to ascending values of the uniform random placement $\pi_U$ corresponds to ordering the set according to a random permutation:

**Lemma 3.5 ([8])** Given the uniform random placement $\pi_U$ on some variable set $V$, we construct a random permutation $\pi'$ on $V$ as follows: Variable $x$ comes before variable $y$ if $\pi_U(x) < \pi_U(y)$. If two variables have the same place, their order is chosen arbitrarily.

Then $\pi'$ is equal in distribution to a random permutation on $V$. 

---

12
3.3. Basic Concepts of The Analysis

**Proof** Two variables have the same place with probability 0, so we can ignore that case. Now by symmetry, the probability of the subset of placements corresponding to a fixed permutation is the same for each permutation.

In the definition of PPZ, the order of the variables is determined by a random permutation. We can replace this random permutation by a uniform random placement, where variables are processed according to ascending place. According to the previous lemma this gives the same result. In the following analysis we do that and we let an element of the sample space in PPZ(F) be the pair of the chosen placement π and the chosen assignment β.

We now define the concept of forced and guessed variables:

**Definition 3.6 ([8])** Consider PPZ(F). We call a variable \( x \in V(F) \) forced if its value was determined by a unit clause. Otherwise, we call \( x \) guessed. For \( \alpha \in \text{sat}(F) \) and a placement \( \pi \) on \( V(F) \), we define \( \text{Forced}(F, \pi, \alpha) \) (\( \text{Guessed}(F, \pi, \alpha) \)) to be the sets of variables that are forced (guessed) in \( \text{Modify}(F, \pi, \alpha) \).

Let \( \beta \) be the assignment chosen in PPZ(F). Then for \( x \in V(F) \), let \( p_F(x, F, \alpha) := \Pr (x \in \text{Forced}(F, \pi_U, \alpha)) \), the probability over the placements that \( x \) is forced in PPZ(F, \( F, \alpha \)).

Note that we define the set of forced and guessed variables not w.r.t. some random assignment, but w.r.t. a fixed satisfying assignment. The following lemma justifies this. It states that if we want that \( \text{Modify} \) returns a fixed satisfying assignment \( \alpha \), then only the guessed variables w.r.t. \( \alpha \) need to be set correctly in the random assignment \( \beta \).

**Lemma 3.7 ([8])** Let \( \alpha \in \text{sat}(F) \), and \( \pi \) be a placement on \( V(F) \). Then for some assignment \( \beta \) on \( V(F) \), \( \text{Modify}(F, \beta, \pi) = \alpha \) if and only if \( \beta |_{\text{Guessed}(F, \pi, \alpha)} = \alpha |_{\text{Guessed}(F, \pi, \alpha)} \).

**Proof** Assume that \( \beta |_{\text{Guessed}(F, \pi, \alpha)} = \alpha |_{\text{Guessed}(F, \pi, \alpha)} \). We do induction on the PPZ-steps. Consider step \( i \) and assume that all previous steps are according to \( \alpha \). Let \( x \) be the variable of step \( i \). If \( x \) is guessed, then the value of \( x \) is chosen according to \( \beta \). It is easily seen that if \( x \) is guessed in this case, then \( x \in \text{Guessed}(F, \pi, \alpha) \) and hence \( \beta(x) = \alpha(x) \). If \( x \) is forced, then any unit clause of \( x \) must be satisfied by \( \alpha \). Hence \( x \) must be chosen according to \( \alpha \).

If \( \beta |_{\text{Guessed}(F, \pi, \alpha)} \neq \alpha |_{\text{Guessed}(F, \pi, \alpha)} \), let step \( i \) be the first step where these assignments differ in the corresponding variable \( x \). By the previous paragraph, we know that all previous steps are according to \( \alpha \). Hence \( x \) is guessed and the value of \( x \) is chosen according to \( \beta \), and the returned assignment is not equal to \( \alpha \) on \( x \).

With these considerations, the probability that PPZ returns a given satisfying assignment \( \alpha \) can now be written as an expectation over the placements of an expression depending on the number of forced variables.
Lemma 3.8 ([8]) Let $\alpha \in \text{sat}(F)$. We have

$$p_{\text{PPZ}}(F, \alpha) = E \left[ 2^{-n(F)} + |\text{Forced}(F, \pi_U, \alpha)| \right].$$

Proof Given $\pi_U$, the probability that $\text{PPZ}(F) = \alpha$ is the proportion of assignments $\beta$ with $\text{MODIFY}(F, \pi_U, \beta) = \alpha$. From Lemma 3.7, we know that $\text{PPZ}(F) = \alpha$ if and only if $\beta$ and $\alpha$ agree on $\text{Guessed}(F, \pi_U, \alpha)$. There are $2^{|\text{Forced}(F, \pi_U, \alpha)|}$ assignments $\beta$ satisfying this. \hfill $\square$

Using Jensen’s inequality and linearity of expectation, the probability that $\text{PPZ}$ returns $\alpha$ can be bounded by an expression depending on the probabilities that the individual variables are forced.

Lemma 3.9 ([8]) Let $\alpha \in \text{sat}(F)$. Then

$$p_{\text{PPZ}}(F, \alpha) \geq 2^{-n(F)} + \sum_{x \in V(F)} p_F(x, F, \alpha).$$

Proof From Lemma 3.8, we have

$$p_{\text{PPZ}}(F, \alpha) = E \left[ 2^{-n(F)} + |\text{Forced}(F, \pi_U, \alpha)| \right].$$

Using Jensen’s inequality (as the exponential function is convex) and linearity of expectation, we obtain

$$p_{\text{PPZ}}(F, \alpha) \geq 2^{-n(F)} + E[|\text{Forced}(F, \pi_U, \alpha)|].$$

It is easily seen that

$$E[|\text{Forced}(F, \pi_U, \alpha)|] = \sum_{V(F)} p_F(x, F, \alpha),$$

and the statement follows. \hfill $\square$

The previous lemmas extend to PPSZ by replacing $F$ by $\text{RESOLVE}(F, s)$. This justifies the following definitions:

Definition 3.10 Let

$$\text{Forced}(F, \pi, \alpha, s) := \text{Forced}(\text{RESOLVE}(F, s), \pi, \alpha),$$

$$\text{Guessed}(F, \pi, \alpha, s) := \text{Guessed}(\text{RESOLVE}(F, s), \pi, \alpha),$$

$$p_F(x, F, \alpha, s) := p_F(x, \text{RESOLVE}(F, s), \alpha).$$

From Lemma 3.9, we get the following corollary for PPSZ:

Corollary 3.11 ([8]) Let $F$ be a CNF and $\alpha \in \text{sat}(F)$. Then

$$p_{\text{PPSZ}}(F, \alpha, s) \geq 2^{-n(F)} + \sum_{x \in V(F)} p_F(x, F, \alpha, s).$$
3.4 Critical Clause Trees

We now want to give a lower bound for \( p_F(x, F, \alpha, s) \), i.e. the probability that a variable is forced given some satisfying assignment \( \alpha \). Given \( \alpha \), it is easily seen that only clauses where \( \alpha \) satisfies exactly one literal might turn into unit clauses. We call such clauses critical:

**Definition 3.12 ([8])** Let \( F \) be a CNF, \( x \in V(F) \) and let \( \alpha \) be an assignment on \( V \). A clause \( C \) is called critical for \( (x, F, \alpha) \) if \( C \in F \), \( \alpha \) satisfies exactly one literal of \( C \), and this literal is over \( x \).

We want to find a representation of the critical clauses for \( (x, \text{Resolve}(F, s), \alpha) \) that we can use to bound \( p_F(x, F, \alpha, s) \). It turns out that we can represent the critical clauses nicely by a rooted tree, where a “cut” corresponds to a critical clause. We call such a tree a critical clause tree. We need to introduce some terminology regarding rooted trees to make this precise.

A rooted tree \( T \) is an acyclic graph with a designated root denoted by \( \text{Root}(T) \). We refer to a vertex of a rooted tree as a node. By \( N(T) \) we denote the set of the nodes of the tree. The depth of a node in a rooted tree is its distance from the root. For a node \( v \), we denote its children by \( \text{Child}(v) \). An internal node is a node that is not a leaf. \( T \) has uniform depth \( d \) if any leaf of \( T \) has depth \( d \). The degree of a node is the number of its children. The subtree of \( T \) rooted at node \( v \) is the tree rooted at \( v \) consisting of \( v \) and the descendants of \( v \).

A set of nodes \( A \) is a cut of \( T \), if it does not include the root and every path from the root to a leaf includes a node of \( A \). A minimal cut is a cut \( A \) where for any \( v \in A \), \( A \setminus \{v\} \) is not a cut. Equivalently, a minimal cut is a cut \( A \) such that no node in \( A \) is an descendant of another node in \( A \). A partial labeling on a rooted tree is a partial function from \( N(T) \) to some set \( V \). We extend the notions of a cut to a set of labels: We say a set of labels \( W \) is a label cut in \( T \), when there is a cut \( A \) in \( T \) s.t. \( L(A) \subseteq W \), i.e. only elements of \( W \) occur as labels in the cut.

**Definition 3.13 ([8])** A rooted tree together with a partial labeling is admissible for some set \( V \), if it has the following properties:

- Each node is either labeled by some element of \( V \) or unlabeled.
- The root is labeled.
- On any path from the root to a leaf, no two nodes have the same label.

A critical clause tree for \( x \) is an admissible tree with the additional property that any cut \( A \) corresponds to a critical clause \( C \) for \( x \), in the sense that the dissatisfied literals of \( C \) must be over variables occurring in the labels of \( A \).
Definition 3.14 ([8]) Let $F$ be a CNF, $x \in V(F)$ and let $\alpha$ be an assignment on $V$.

A rooted tree with a partial labeling $L$ is a critical clause tree for $(x, F, \alpha)$, if the tree is admissible for $V(F)$ and has the following additional properties:

- The label of the root is $x$.
- For any cut $A$ of the tree, there is a critical clause $C$ for $(x, F, \alpha)$ with $V(C) \subseteq L(A) \cup \{x\}$ (remember that $L(A)$ denotes the image of $A$ under $L$). Equivalently, for any label cut $W$, there is a critical clause $C$ for $(x, F, \alpha)$ with $V(C) \subseteq W \cup \{x\}$.

We are now ready to prove a lemma that states that for any satisfiable $(\leq k)$-CNF and a sufficiently isolated satisfying assignment, there exists a “good” critical clause tree for all variables after bounded resolution. Afterwards we will show how to bound the probability that a variable is forced using such a critical clause tree. We first give some intuition on how the trees are built and why they are actual critical clause trees, together with an example. Then we give a formal proof.

Let $F$ be a $(\leq k)$-CNF that is satisfied only by the all-one assignment $\alpha$. To build a critical clause tree, we start with a node labeled $x$. Now we repeatedly extend a leaf node $v$ by at most $k - 1$ children as follows: Let $L$ be the set of labels occurring on the path from the root to $v$. At the beginning, $L = \{x\}$. As $\alpha$ is the only satisfying assignment, $\alpha \oplus L$ does not satisfy $F$, and there is a clause $C$ not satisfied by $\alpha \oplus L$. For each negative literal of $C$ we extend $v$ by a child labeled with the variable the literal is over. If there are no negative literals in $C$, we extend $v$ by an unlabeled child. This process can be repeated indefinitely. Note that we can always extend an unlabeled leaf by an unlabeled child. As the labels along a path from the root to a leaf are unique, all nodes at depth at least $n(F)$ will be unlabeled. If every leaf has depth at least $n(F)$, then there is a cut consisting only of unlabeled nodes; this corresponds to the fact that by unrestricted resolution the clause $\{x\}$ can be obtained. However, as we consider $s(n(F))$-bounded resolution, we cannot add too many nodes to the tree (remember that $s$ is a slowly growing function). We are on the safe side if the tree consists of at most $\lceil s(n(F)) \rceil$ nodes. With this, we can construct a tree of uniform depth $d := \lceil \log_k(s(n(F))) \rceil$. Note that $d$ goes to $\infty$ as $n(F)$ goes to $\infty$.

To show that the constructed tree is indeed a critical clause tree, we have to show the existence of the required critical clause $C$ for every cut $A$. This is shown by giving a resolution deduction using clauses used to create nodes on a path from a node in $A$ to the root, starting from the nodes with largest depth. A case distinction would be needed because some clauses will not be needed in the resolution. In the proof we will avoid the case distinction by making use of the stronger/weaker relation on CNFs.
We give a simple example: Let
\[ F := \{ \{x, \overline{y}, z\}, \{x, y, \overline{a}\}, \{z, \overline{b}, \overline{c}\}, \{x, z, c\}\}. \]

For the all-one assignment and \( x \), we can get the tree shown in Figure 3.1 by the described procedure. The two nodes labeled \( a \) and \( b \) together with the unlabeled node form a cut in this tree. Hence \( \{a, b\} \) is a label cut. We have \( R(\{z, \overline{b}, \overline{c}\}, \{x, z, c\}) = \{x, z, \overline{b}\} \), \( R(\{x, y, \overline{a}\}, \{x, y, \overline{a}\}) = \{x, \overline{a}, \overline{b}\} \) and \( R(\{x, z, \overline{b}\}, \{x, z, \overline{a}\}) = \{x, \overline{a}, \overline{b}\} \), giving the required critical clause.

Now we come to the formal proof. It is easy to see that for a tree of uniform depth \( d \), it is sufficient for an assignment to be \( d \)-isolated given flipped \( x \) (see Definition 2.1).

**Lemma 3.15** Let \( F \) be a satisfiable \((\leq k)\)-CNF, \( x \in V(F) \). Let \( \alpha \in \text{sat}(F) \) be \( d \)-isolated given flipped \( x \). If \( s \geq k^d \), then there exists a critical clause tree for \((x, \text{Resolve}(F, s), \alpha)\) of uniform depth \( d \) and maximum degree \( k - 1 \).

**Proof** Let \( \alpha \) be w.l.o.g. the all-one assignment. The tree \( T \) with labeling \( L \) is constructed as follows: Start with a node with label \( x \) as root. As long as there is a leaf \( v \) with depth less than \( d \), do the following:

Let \( P(v) \) be the set of nodes on the path from \( v \) to the root (both inclusive). As \( x \in L(P(v)) \) and \( |L(P(v))| \leq d \), we have \( \alpha \oplus P(v) \notin \text{sat}(F) \). Let \( C \) be a clause in \( F \) not satisfied by \( \alpha \oplus P(v) \). This means that the positive literals of \( C \) are in \( L(P(v)) \), and the negative literals of \( C \) are not over variables in \( L(P(v)) \). \( C \) has at least one positive literal (as otherwise \( \alpha \) would not satisfy \( C \)) and hence at most \( k - 1 \) negative literals. For each negative literal of \( C \), add a child to \( v \), labeled with the corresponding variable. If there are no negative literals in \( C \), add an unlabeled child to \( v \).

It follows from the construction that \( T \) is admissible for \( V(F) \), has uniform depth \( d \) and maximum degree \( k - 1 \). It remains to prove that \( T \) is a critical clause tree for \((x, \text{Resolve}(F, s), \alpha)\). Trivially, the label of the root is \( x \).

We need to show the existence of the required critical clause for every cut \( A \). We first annotate each internal node \( v \) by the clause \( C(v) := L(P(v)) \cup L(\text{Child}(v)) \), i.e. the clause obtained by combining the labels on the path from \( v \) to the root with the negated labels of the children of \( v \). Let \( F' \) be the set of all clauses \( C(v) \) for an internal node \( v \) of \( T \). It follows from the construction that \( F \) is stronger than \( F' \).

We will show that for any minimal cut \( A \) there is a resolution deduction of \( C(A) := \overline{L(A)} \cup \{x\} \) from \( F' \). This is sufficient: As \( F' \) is only over variables...
that appear as labels in the tree and it is easily seen that $T$ contains at most $k^d$

nodes, the resolution deduction is $k^d$-bounded. Lemma 2.3 then tells us that

there is a $k^d$-bounded resolution deduction of a subclause $C'(A)$ of $C(A)$.

As $\alpha$ satisfies $C'(A)$, we know that $x \in C'(A)$. Furthermore, $C'(A)$ satisfies

the required property for any cut $A'$ with $A \subseteq A'$.

Let $A$ be a minimal cut of $T$. We show that there is a resolution deduction of $C(A)$ from $F$. Let $B$ be the set of nodes not in $A$ on the paths from nodes of $A$ to the root. We can order the nodes in $B$ s.t. each node comes after its children. For $v \in B$, let $D(v)$ be the clause consisting of $L(P(v))$ and the negated labels of nodes in $A$ that are descendants of $v$. We show by induction along this order that for every $v \in B$, there is a resolution derivation of $D(v)$ from $F'$. As $C(A) = D(\text{Root}(T))$, this proves the statement.

Consider $v \in B$. If all children of $v$ are in $A$, then as $A$ is a minimal cut, we have $D(v) = C(v)$. If not, we know that all children of $v$ are in $B$ (as otherwise $A$ would not be a minimal cut) and come before in the induction. We can resolve $C(v)$ incrementally with $D(v')$ for each child $v'$ of $v$. It is easy to see that this is possible and that it results in $D(v)$.

We have now proven that in the unique case, critical clause trees up to some depth increasing with $n(F)$ exists for $\text{Resolve}(F, s(n(F)))$. Now we want to use this to bound $p_F(x, F, \alpha, s(n(F)))$. Let $T$ be a critical clause tree for $x$. The key observation is that if the variables corresponding to some cut of $T$ come before $x$ in $\text{Modify}$, then $x$ is forced. With this, it remains to bound the probability that the labels of some cut of $T$ come before $x$ w.r.t. some random placement (we will need non-uniform random placements in the next chapter). For this we first need to show a correlation inequality which tells us essentially that the worst case occurs if all the labels of the tree are distinct. Using this inequality, we can give a recursive formula that bounds the probability that we have such a cut. This inequality can then be solved to bound the probability for an idealized tree with infinite depth. We then want to show that the probability of such a cut in trees with finite depths converges to the probability of an infinite tree if $d$ goes to $\infty$. The convergence itself is not so hard to prove using measure theoretic methods. However, we will use elementary analysis to prove the convergence as in [8]; this is tedious but it gives us a bound on the convergence rate.

Now consider an admissible tree $T$ for $V(F)$. For $x \in V(F)$ and a placement $\pi$ on $V(F)$, we define $\text{Before}(x, \pi) := \{ y \in V(F) : \pi(y) < \pi(x) \}$, the set of variables that come before $x$ in $\pi$. For $r \in [0,1]$ and a placement $\pi$ on $V(F)$, we define $\text{Before}(r, \pi) := \{ y \in V(F) : \pi(y) < r \}$, the set of variables that have place less than $r$.

Let $x := L(\text{Root}(T))$, the label of the root of $T$. We define $\text{Cut}(T)$ as the set of placements $\pi$ on $V(F)$ s.t. $\text{Before}(x, \pi)$ is a label cut in $T$. We define
Cut\((T, r)\) as the set of placements \(\pi\) on \(V(F)\) s.t. \(\text{Before}(r, \pi) \setminus \{x\}\) is a label cut in \(T\). The second definition seems unnecessary; it essentially ignores \(x\) and \(\pi(x)\). However, we will need this in Lemma 3.18, where we consider subtrees for which the cut will no longer be w.r.t. the label of the root.

Let \(\pi\) now be a random placement on \(V(F)\) according to some given distribution. We define \(Q(T, \pi) := \Pr(\pi \in \text{Cut}(T))\). We define \(Q(T, r, \pi) := \Pr(\pi \in \text{Cut}(T, r))\). If the random placement \(\pi\) we consider is clear from the context, we write \(Q(T)\) for \(Q(T, \pi)\) and \(Q(T, r)\) for \(Q(T, r, \pi)\). If \(\pi(x)\) is chosen independently and u.a.r. from \([0, 1]\), then we have by the law of total probability

\[
Q(T, \pi) = \int_0^1 Q(T, r, \pi)dr.
\]

The following lemma connects the probability that a variable is forced with the probability that \(\text{Before}(x, \pi)\) is a label cut in its critical clause tree:

**Lemma 3.16** ([8]) If \(T\) is a critical clause tree for \((x, \text{RESOLVE}(F, s), \alpha)\), then

\[
p_F(x, F, \alpha, s) \geq Q(T, \pi_U).
\]

**Proof** We show that if \(\text{Before}(x, \pi)\) is a label cut in \(T\), then \(x\) is forced in \(\text{MODIFY(RESOLVE(F, s), \pi, \alpha)}\). Assume \(\text{Before}(x, \pi)\) is a label cut in \(T\). By the definition of a critical clause tree, there is a clause \(C \in \text{RESOLVE}(F, s)\) that is critical for \((x, F, \alpha)\) with \(V(C) \subseteq \text{Before}(x, \pi) \cup \{x\}\). It is now easily seen that before the step of \(x\) in \(\text{MODIFY(RESOLVE(F, s), \pi, \alpha)}\), \(C\) has become a unit clause for \(x\) that forces \(x\).

To bound \(Q(T, r, \pi)\), we need to prove a technical lemma that is a special case of the FKG inequality [2]. We use this inequality to show that the worst case of \(Q(T, r, \pi)\) is when all labels of \(T\) are distinct.

**Lemma 3.17** Let \(n \in \mathbb{N}_0\), and let \(S := 2^n\). Let \(f\) and \(g\) be functions from \(S\) to the non-negative real numbers with the property that for all \(B' \subseteq B \in S\) we have \(f(B') \leq f(B)\) and \(g(B') \leq g(B)\) (i.e. \(f\) and \(g\) are monotone increasing). Form the random set \(A \subseteq [n]\) by including each \(i \in [n]\) with probability \(p(i)\), independently.

Then we have

\[
E[f(A)]E[g(A)] \leq E[f(A)g(A)].
\]

**Proof** The proof is by induction on \(n\). For \(n = 0\), the statement is trivial. Assume the statement holds for \(n' = n - 1\). We show it for \(n\). We have to show

\[
E[f(A)]E[g(A)] \leq E[f(A)g(A)].
\]

Let \(f_0 := f(A \setminus \{n\}), f_1 := f(A \cup \{n\}), g_0 := g(A \setminus \{n\})\) and \(g_1 := g(A \cup \{n\})\). Let \(p := p(n)\) and \(q := 1 - p(n)\). With this,

\[
E[f(A)]E[g(A)] \leq q^2E[f_0]E[g_0] + pqE[f_0]E[g_1] + pqE[f_1]E[g_0] + p^2E[f_1]E[g_1].
\]
3. General Properties and Unique SAT

It is easy to see that we can use the induction hypothesis for \( n - 1 \) to obtain

\[
E[f(A)]E[g(A)] \leq q^2E[f_0g_0] + pqE[f_0g_1] + pqE[f_1g_0] + p^2E[f_1g_1]
\]

\[
= E[q^2f_0g_0 + pqf_0g_1 + pqf_1g_0 + p^2f_1g_1].
\]

Now as \( pq(f_1 - f_0)(g_1 - g_0) \geq 0 \), we have

\[
pqf_0g_1 + pqf_1g_0 \leq pqf_0g_0 + pqf_1g_1.
\]

With this and using \( q^2 + pq = q \) and \( pq + p^2 = p \), we get

\[
E[f(A)]E[g(A)] \leq E[q^2f_0g_0 + pqf_0g_1 + pqf_1g_0 + p^2f_1g_1]
\]

\[
= E[qf_0g_0 + pf_1g_1] = E[f(A)g(A)].
\]

With the previous lemma we can now bound \( Q(T, r, \pi) \) in terms of \( Q(T', r, \pi) \) for subtrees \( T' \) of \( T \). A subtree \( T' \) of an admissible tree \( T \) is admissible if and only if the root of \( T' \) is labeled. Remember that nice random placements are placements \( \pi \) where for every variable \( x \) the place \( \pi(x) \) is chosen independently.

**Lemma 3.18 ([8])** Let \( \pi \) be a nice random placement on some set \( V \). Let \( T \) be an admissible tree with at least two nodes. Let \( x := L(\text{Root}(T)) \). Let \( r \in [0, 1] \). Let \( T_1, T_2, \ldots, T_l \) be the (admissible) subtrees of \( T \) rooted at those children of the root of \( T \) that are labeled; let \( x_i := L(\text{Root}(T_i)) \) and let \( p_i \) be the probability that \( \pi(x_i) \leq r \). Then

\[
Q(T, r, \pi) \geq \prod_{i=1}^{l} (p_i + (1 - p_i)Q(T_i, r, \pi)).
\]

If \( T \) consists of exactly one node, we trivially have \( Q(T, r, \pi) = 0 \) and \( Q(T, \pi) = 0 \).

**Proof** As \( x \) only occurs as label of the root in \( T \) and as \( \pi \) is a nice random placement, it is sufficient to consider only placements with \( \pi(x) = r \). With this, \( \text{Cut}(T) \) is easily seen to be the intersection of the events \( K_i := \text{Cut}(T_i, r) \cup \{ \pi : \pi(x_i) \leq r \} \). As \( \pi \) is a nice placement and \( x \) does not appear in \( T_i \), we have \( \Pr(\text{Cut}(T_i, r)) \leq Q(T_i, r) \). Also, \( \text{Cut}(T_i, r) \) and \( \{ \pi : \pi(x_i) \leq r \} \) are independent events, so \( \Pr(K_i) = p_i + (1 - p_i)Q(T_i, r) \). Let \( W := W(\pi, r) \) be the set of variables \( y \) with \( \pi(y) < r \). Each event \( K_i \) depends only on \( W_i \); let \( K_i(W) \) be the indicator function of \( K_i \) and let \( K(W) \) be the indicator function of \( K \). It is clear that for \( W' \subseteq W \) we have \( K_i(W') \leq K_i(W) \). Also, \( K(W) = \prod_{i=1}^{l} K_i(W) \). We have \( Q(T, r) = \sum_{U \subseteq V} \Pr(W = U)K(W) \), and \( \Pr(K_i) = \sum_{U \subseteq V} \Pr(W = U)K_i(W) \). It is now easily seen that \( Q(T, r) \geq \prod_{i=1}^{l} \Pr(K_i) \) follows from Lemma 3.17 with induction on \( t \). \( \square \)
Assume now that $T$ has infinite depth and that $\pi$ is the uniform random placement. Here it is easily seen that the worst case is if $T$ has $k - 1$ children and all nodes are labeled (for this, we assume there is an infinite supply of variables). In this case each child of $T$ is isomorphic to $T$ itself and from Lemma 3.18, we get the equation $Q(T, r, \pi) = (r + (1 - r)Q(T_i, r, \pi))^{k-1}$. This motivates the following definition:

**Definition 3.19 ([8])**

Let $f_k(t, r) := (r + (1 - r)t)^{k-1}$. Let $R_k(r)$ be the smallest $t \geq 0$ that satisfies $f_k(t, r) = t$. This is well-defined, as trivially $f_k(1, r) = 1$ and $f_k(t, r) - t$ is a polynomial in $t$.

We define $R_k := \int_0^1 R_k(r) dr$.

$R_k$ can be seen as the probability that a variable is forced assuming it has an infinite critical clause tree as described before. For $k = 3$, we will show $R_3(r) = \min \{ (\frac{r}{1-r})^2, 1 \}$. Now we consider finite trees. We can bound the cut probability for a tree with maximum degree $k - 1$ and uniform depth $d$ by a recursive expression. After that we want to show that for large $d$, this probability converges to $R_k$.

**Definition 3.20 ([8])**

Let $Q_k(0, r) := 0$. For an integer $d > 0$, let $Q_k(d, r) := f_k(Q_k(d - 1, r), r)$. Let $Q_k(d) := \int_0^1 Q_k(d, r) dr$.

Using Lemma 3.18, we observe:

**Observation 3.21**

If $T$ has uniform depth $d$ and maximum degree $k - 1$, we have by Lemma 3.18 and induction on $d$

$$Q_k(T, r) \geq Q_k(d, r),$$

and hence

$$Q_k(T) \geq Q(d).$$

It remains to show that $Q_k(d)$ converges to $R_k$ for large $d$. The convergence itself is not so hard to see using measure theoretic methods. We will use elementary analysis which is not as elegant but additionally bounds the rate of convergence.

### 3.5 Analysis for Unique 3-SAT

In this section we only consider unique 3-SAT. This is much easier than unique $k$-SAT, which we will analyze in the next section. Let $F$ be a satisfiable ($\leq 3$)-CNF. In this section, we omit the subscript $k$ and write $R, f, Q$ instead of $R_3, f_3, \text{and } Q_3$.

Let $\varepsilon(d) := \frac{1}{d+1}$. In this section, we show the following lemma:
Lemma 3.22 ([8]) Let $F$ be a satisfiable $(\leq 3)$-CNF, $\alpha \in \text{sat}(F)$ and $x \in V(F)$. If there exists a critical clause tree for $(x, F, \alpha)$ of uniform depth $d$ and maximum degree $k - 1$, then
\[ p_F(x, F, \alpha, 3^d) \geq 2 - 2\ln 2 - \epsilon(d). \]

Lemma 3.22 together with Lemma 3.15 and Corollary 3.11 imply the following theorem:

**Theorem 3.23** If $F$ is a satisfiable $(\leq 3)$-CNF with a $\log_3(s(n(F)))$-isolated satisfying assignment, we have
\[ p_{\text{PPSZ}}(F, \alpha, s(n(F))) \geq 2^{-2(2\ln 2 - 1)n(F) - o(n(F))}. \]

There is a randomized algorithm for unique-3-SAT with running time
\[ O(2^{2(2\ln 2 - 1)n(F) + o(n(F))}) \approx O(1.30704^n(F)). \]

**Proof** As $s$ is a slowly growing function, $\epsilon(\log_3 s(n(F))) = o(1)$. As PPSZ runs in time $2^{o(n(F))}$, we can construct the required algorithm by repeatedly running PPSZ. 

Now we prove Lemma 3.22. First we give a closed expression for $R(r)$:

**Lemma 3.24 ([8])** For $r \in [\frac{1}{2}, 1]$, we have $R(r) = 1$.

For $r \in [0, \frac{1}{2}]$, we have $R(r) = \left(\frac{r}{1-r}\right)^2$.

For $r \in [0, 1]$ and $t \leq R(r)$, we have $f(t, r) \leq R(r)$.

**Proof** Obviously we have $f(1, r) = 1$ for all $r \in [0, 1]$, hence $R(r) \leq 1$. For $r \in [\frac{1}{2}, 1]$ and $t < 1$, we have
\[ f(t, r) = (r + (1-r)t)^2 \geq \left(\frac{1}{2} + \frac{1}{2}t\right)^2 = t + \left(\frac{1}{2} - \frac{1}{2}t\right)^2 > t. \]

This proves the first statement.

For $r \in [0, \frac{1}{2}]$, we have
\[ f\left(\left(\frac{r}{1-r}\right)^2, r\right) = \left(\frac{r}{1-r} + \frac{r^2}{1-r}\right)^2 = \left(\frac{r}{1-r}\right)^2. \]

Furthermore, for fixed $r \in [0, \frac{1}{2}]$ and $t < \left(\frac{r}{1-r}\right)^2$, let $g(t) := f(t, r) - t$. We have $g(t) = (r + (1-r)t)^2 - t$ and
\[ g'(t) = (1-r)2(r + (1-r)t) - 1 = 2(1-r)^2t + 2r(1-r) - 1 < 2r^2 + 2r(1-r) - 1. \]
3.5. Analysis for Unique 3-SAT

As for \( r \leq \frac{1}{2}, \) we have \( r^2 \leq \frac{1}{4} \) and \( r(1 - r) \leq \frac{1}{4}, \) this is at most 0. This implies that for \( t < \left( \frac{1}{1 - r} \right)^2, \) we have \( f(t, r) > t. \) The second statement follows immediately.

To prove the third statement, it is enough to check that the derivative of \( f(t, r) \) w.r.t. \( t \) is non-negative for \( r \in [0, 1] \) and \( t \geq 0. \) We have

\[
\frac{\partial f(t, r)}{\partial t} = (1 - r)2(r + (1 - r)t) \geq 0.
\]

\( \square \)

By elementary integration, we obtain \( R = 2 - 2\ln 2. \)

We now have to show that \( Q(d, r) \) converges to \( R(r). \)

**Definition 3.25 (8)** Let \( \Delta(d, r) := R(r) - Q(d, r) \) and let \( \Delta(d) = R - Q(d). \)

The following lemma implies Lemma 3.22 together with Observation 3.21 and Lemma 3.16:

**Lemma 3.26 (8)** For \( d \geq 1, \) we have \( 0 \leq \Delta(d) \leq \varepsilon(d). \)

**Proof** \( 0 \leq \Delta(d, r) \) follows from the third statement of Lemma 3.24 with induction on \( d; \) hence we also have \( 0 \leq \Delta(d). \) We have to show that \( \Delta(d) \leq \varepsilon(d). \)

We now fix \( r \in [0, 1]. \) For \( d > 0, \) we have

\[
\Delta(d, r) = R(r) - Q(d, r) = R(r) - f(Q(d - 1, r), r)
\]

\[
= R(r) - f(R(r) - \Delta(d - 1, r), r).
\]

We show that for any \( \gamma \in [0, R(r)], \)

\[
0 \leq R(r) - f(R(r) - \gamma, r) \leq \gamma \cdot 2(1 - r)(r + (1 - r)R(r)).
\]

The first inequality follows from the third statement of Lemma 3.24. For the second inequality, we have

\[
R(r) - f(R(r) - \gamma, r) = f(R(r), r) - f(R(r) - \gamma, r)
\]

\[
= (r + (1 - r)R(r))^2 - (r + (1 - r)(R(r) - \gamma))^2.
\]

Using \( s^2 - t^2 = (s - t)(s + t), \) this is

\[
= \gamma(1 - r)(2r + (2 - 2r)R(r) - (1 - r)\gamma)
\]

\[
\leq \gamma \cdot 2(1 - r)(r + (1 - r)R(r)).
\]

By induction on \( d \) with \( \gamma = \Delta(d - 1, r), \) we obtain

\[
\Delta(d, r) \leq (2(1 - r)(r + (1 - r)R(r)))^d.
\]

23
3. General Properties and Unique SAT

Hence
\[ \Delta(d) \leq \int_0^1 (2(1 - r) (r + (1 - r)R(r)))^d dr. \]

For \( r \in [\frac{1}{2}, 1] \), we have \( R(r) = 1 \), and
\[ 2(1 - r) (r + (1 - r)R(r)) = 2(1 - r). \]

For \( r \in [0, \frac{1}{2}] \), we have
\[
2(1 - r) (r + (1 - r)R(r)) = 2(1 - r) \left( r + \left( 1 - r \right) \frac{r}{1 - r} \right) = 2(1 - r) \left( \frac{r}{1 - r} \right) = 2r.
\]

Hence
\[
\Delta(d) \leq \int_0^{\frac{1}{2}} (2r)^d dr + \int_{\frac{1}{2}}^1 (2(1 - r))^d dr = \frac{1}{d + 1}. \]

3.6 Analysis for Unique k-SAT

Now we show the analysis for unique \( k \)-SAT from [8]. We generalize the last section to arbitrary \( k \geq 3 \). Later in the thesis, the statements of this section are referenced. For \( k = 3 \), the corresponding statements of the previous section can be used equivalently.

Let \( \epsilon_k(d) := \frac{3}{(d - 1)(k - 2) + 2} \). In this section, we show the following lemma:

Lemma 3.27 ([8]) Let \( F \) be a satisfiable \((\leq k)\)-CNF, \( \alpha \in \text{sat}(F) \) and \( x \in V(F) \).
If there exists a critical clause tree for \((x, F, \alpha)\) of uniform depth \( d \) and maximum degree \( k - 1 \), then
\[ p_F(x, F, \alpha, k^d) \geq R_k - \epsilon(d, k). \]

Note that \( \epsilon_3(d) = \frac{3}{d+1} > \epsilon(d) = \frac{1}{d+1} \). This is because we could give a better analysis when specifically considering 3-SAT.

We define \( S_t(t) \); it will turn out to be the inverse of \( R_k(r) \) for \( r \in [0, \frac{k-2}{k-1}] \). We need this as we cannot give a closed expression of \( R_k(r) \) for general \( k \).
3.6. Analysis for Unique k-SAT

<table>
<thead>
<tr>
<th>$k$</th>
<th>$S_k$</th>
<th>$2^S_k$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.3862943612</td>
<td>1.307031908</td>
</tr>
<tr>
<td>4</td>
<td>0.5548181152</td>
<td>1.468983423</td>
</tr>
<tr>
<td>5</td>
<td>0.6502378685</td>
<td>1.569426938</td>
</tr>
<tr>
<td>6</td>
<td>0.7118242317</td>
<td>1.637873836</td>
</tr>
<tr>
<td>7</td>
<td>0.7549118405</td>
<td>1.687528468</td>
</tr>
<tr>
<td>8</td>
<td>0.7867645774</td>
<td>1.725201144</td>
</tr>
</tbody>
</table>

Table 3.1: Numerical Values of $S_k$ and $2^S_k$ for Small $k$

Definition 3.28 For $t \in [0,1)$, let $S_k(t) := \frac{1}{1-t}$. Let $S_k(1) := \frac{k-2}{k-1}$. Let $S_k := \int_0^1 S_k$.

$R_k$ and $S_k$ are directly related.

Lemma 3.29 ([8])

$$R_k = 1 - S_k.$$ 

While $R_k$ is the probability that a variable is forced (in an infinite tree), $S_k$ can be seen as the probability that a variable is guessed. We will prove this lemma later.

Lemma 3.27 and Lemma 3.29 together with Lemma 3.15 and Corollary 3.11 imply the following theorem:

Theorem 3.30 ([8]) If $F$ is a satisfiable ($\leq k$)-CNF with a $\log_k(s(n(F)))$-isolated satisfying assignment, we have

$$p_{PPSZ}(F, \alpha, s(n(F))) \geq 2^{-S_k n(F) - o(n(F))}.$$ 

There is a randomized algorithm for unique-$k$-SAT with running time $O\left(2^{S_k n(F) + o(n(F))}\right)$.

Proof As $s$ is a slowly growing function, $\varepsilon_k(\log_k s(n(F))) = o(1)$. As PPSZ runs in time $2^{o(n(F))}$, we can construct the required algorithm by repeatedly running PPSZ.

We will see in the next chapter that for $k \geq 5$, the same bound holds for general k-SAT. For small $k$, the numerical values of $S_k$ and $2^S_k$ are provided in Table 3.1.

Now we prove Lemma 3.27. We need the following facts that are easily proved by induction. For $n \in \mathbb{N}$ and $p, q \in \mathbb{R}$, we have

$$p^n - q^n = (p - q) \sum_{i=0}^{n-1} p^i q^{i-1}. \quad (3.1)$$
3. General Properties and Unique SAT

\[ 1 - (n + 1)q^n + nq^{n+1} = (1 - q)^2 \sum_{i=0}^{n-1} (i+1)q^i. \]  \hspace{1cm} (3.2)

If \( p \neq 0 \) and \( n \geq 2 \), we have

\[ (1 + p)^n > 1 + np. \]  \hspace{1cm} (3.3)

**Lemma 3.31** ([8]) For \( t \in [0, 1] \), we have

\[ S_k(t) = 1 - \frac{1}{\sum_{i=0}^{k-2} t^{i+1}}. \]

Furthermore, \( S_k(t) \) is a strictly increasing, continuously differentiable function mapping \([0, 1]\) to \([0, \frac{k-2}{k-1}]\).

**Proof** We have for \( t \in [0, 1) \)

\[ S_k(t) = \frac{\frac{1}{1-t} - t}{1-t} = 1 - \frac{\frac{1}{1-t}}{1-t}. \]

Using (3.1) with \( p := 1 \) and \( q := \frac{1}{1-t} \), this is, as \( t \neq 1 \)

\[ = 1 - \frac{1}{\sum_{i=0}^{k-2} t^{i+1}}. \]

For \( t = 1 \), we have

\[ 1 - \frac{1}{\sum_{i=0}^{k-2} t^{i+1}} = 1 - \frac{1}{k-1} = \frac{k-2}{k-1}. \]

The properties of \( S_k(t) \) are now easily seen. \( \square \)

We can characterize \( S_k \) by an infinite sum. We need this lemma in the next chapter for \( k \geq 5 \).

**Lemma 3.32** ([8]) \( S_k = 1 - \frac{1}{k-1} \sum_{j=1}^{\infty} \frac{1}{j(j+1)} \).

**Proof** First we see for \( t \in [0, 1) \) that

\[ 1 - S_k(t) = \frac{1 - t^{\frac{1}{k}}}{1-t} = \left(1 - t^{\frac{1}{k}}\right) \sum_{i=0}^{\infty} t^i \]

\[ = \sum_{i=0}^{\infty} \left( t^i - t^{i+\frac{1}{k}} \right). \]
Hence we have

\[ S_k = 1 - \int_0^1 \sum_{i=0}^{\infty} \left( t^i - t^{i+\frac{1}{k-1}} \right) dt. \]

As we have \( t^i - t^{i+\frac{1}{k-1}} \geq 0 \) for \( t \in [0, 1] \), we can use the monotone convergence theorem to obtain

\[ S_k = 1 - \sum_{i=0}^{\infty} \left( \frac{1}{i+1} - \frac{1}{i+1 + \frac{1}{k-1}} \right) = 1 - \sum_{j=1}^{\infty} \left( \frac{1}{j} - \frac{1}{j + \frac{1}{k-1}} \right) \]

\[ = 1 - \sum_{j=1}^{\infty} \frac{1}{k-1} \left( \frac{1}{j + \frac{1}{k-1}} \right) = 1 - \frac{1}{k-1} \sum_{j=1}^{\infty} \frac{1}{j + \frac{1}{k-1}}. \]

□

Now we characterize \( R_k(r) \):

**Lemma 3.33 ([8])** For \( r \in \left[ \frac{k-2}{k-1}, 1 \right] \), we have \( R_k(r) = 1 \).

For \( r \in [0, \frac{k-2}{k-1}] \), \( R_k(r) \) is the inverse of \( S_k(t) \).

For \( r \in [0, 1] \) and \( t \leq R_k(r) \), we have \( f_k(r, t) \leq R_k(r) \).

**Proof** We have \( f(1, r) = 1 \) for all \( r \in [0, 1] \), hence \( R_k(r) \leq 1 \). For \( r \in \left[ \frac{k-2}{k-1}, 1 \right] \) and \( t < 1 \), we have

\[ f_k(t, r) = (r + (1-r)t)^{k-1} \geq \left( \frac{k-2}{k-1} + \frac{1}{k-1} t \right)^{k-1} \]

\[ = \left( 1 - \frac{1}{k-1} (1-t) \right)^{k-1} > t. \]

The last inequality follows from (3.3). This proves the first statement.

For \( t \in [0, 1) \), the following statements are equivalent:

\( (r + (1-r)t)^{k-1} = t. \)

\( r + (1-r)t = t^{\frac{k}{k-1}}. \)

\( r(1-t) = t^{\frac{1}{k-1}} - t. \)

\( r = \frac{t^{\frac{1}{k-1}} - t}{1-t} = S_k(t). \)

Hence if \( R_k(r) < 1 \), then \( S_k(t) \) is the inverse of \( R_k(r) \). We have \( R_k(0) = 0 \) because of \( f_k(0, 0) = 0 \), and by Lemma 3.31, \( S_k(t) \) is a strictly increasing
3. General Properties and Unique SAT

continuous function from $[0, 1]$ to $[0, \frac{k-2}{k-1}]$. It follows that $R_k(r)$ is its inverse from $[0, \frac{k-2}{k-1}]$ to $[0, 1]$.

To prove the third statement, it is enough to check that the derivative of $f_k(t, r)$ w.r.t. $t$ is non-negative for $r \in [0, 1]$ and $t \geq 0$. We have

$$\frac{\partial f_k(t, r)}{\partial t} = (1 - r)(k - 1)(r + (1 - r)t)^{k - 2} \geq 0.$$  \hfill \Box

We can now prove Lemma 3.29:

**Proof (Lemma 3.29)** We have

$$1 - S_k = 1 - \int_0^1 S_k(t) dt$$

By integration by parts, this is

$$= 1 - 1 \cdot S_k(1) + 0 \cdot S_k(0) + \int_0^1 S_k'(t) t dt = \frac{1}{k - 1} + \int_0^1 R_k(S_k(t)) S_k'(t) dt.$$  

Using the substitution $r = S_k(t)$, this is

$$= \frac{1}{k - 1} + \int_0^{1 - \frac{r}{k - 1}} R_k(r) dr = \int_0^1 R_k(r) dr = R_k.$$  \hfill \Box

Finally, we have to show that $Q_k(d, r)$ converges to $R_k(r)$. This is the most complex part.

**Definition 3.34 ([8])** Let $\Delta_k(d, r) := R_k(r) - Q_k(d, r)$, let $\Delta_k(d) = R_k - Q_k(d)$. The following lemma implies Lemma 3.27 together with Observation 3.21 and Lemma 3.16:

**Lemma 3.35 ([8])** For $d \geq 1$, we have $0 \leq \Delta_k \leq \varepsilon_k$.

**Proof** $0 \leq \Delta_k(d, r)$ follows from the third statement of Lemma 3.33 with induction on $d$; hence we also have $0 \leq \Delta_k(d)$. We have to show that $\Delta_k(d) \leq \varepsilon_k(d)$.

We now fix $r \in (0, 1]$. For $d > 0$, we have

$$\Delta_k(d, r) = R_k(r) - Q_k(d, r) = R_k(r) - f_k(Q(d - 1, r), r)$$

$$= R_k(r) - f_k(R_k(r) - \Delta_k(d - 1, r), r).$$

We show that for any $\gamma \in [0, R_k(r)]$, we have

$$0 \leq R_k(r) - f_k(R_k(r) - \gamma, r) \leq \gamma \cdot \frac{(k - 1)(1 - r)R_k(r)}{r + (1 - r)R_k(r)}.$$
3.6. Analysis for Unique k-SAT

The first inequality follows the third statement of Lemma 3.33. For the second inequality, we have

\[ R_k(r) - f_k(R_k(r) - \gamma, r) = f_k(R_k(r), r) - f_k(R_k(r) - \gamma, r) \]

\[ = (r + (1 - r)R_k(r))^{k-1} - (r + (1 - r)(R_k(r) - \gamma))^{k-1}. \]

Using (3.1), this is

\[ = \gamma(1 - r) \sum_{j=0}^{k-2} (r + (1 - r)R_k(r))^{k-2-j} (r + (1 - r)(R_k(r) - \gamma))^j \]

\[ \leq \gamma(k - 1)(1 - r) (r + (1 - r)R_k(r))^{k-2} \]

As \( f_k(R_k(r), r) = R_k(r) \), this is

\[ = \gamma(k - 1) \frac{(1 - r)R_k(r)}{r + (1 - r)R_k(r)}. \]

By induction on \( d \) with \( \gamma = \Delta_k(d - 1, r) \), we obtain

\[ \Delta_k(d, r) \leq \left( \frac{(k - 1)(1 - r)R_k(r)}{r + (1 - r)R_k(r)} \right)^d. \]

Hence

\[ \Delta_k(d) \leq \int_0^1 \left( \frac{(k - 1)(1 - r)R_k(r)}{r + (1 - r)R_k(r)} \right)^d dr. \]

We split the range of integration in two parts. For \( r \in \left[ \frac{k-2}{k-1}, 1 \right] \), we have \( R_k(r) = 1 \). The integral over this range is easily calculated to be \( \frac{1}{(k-1)(d+1)} \leq \frac{1}{(k-2)(d-1)+2} \)

For \( r \in (0, \frac{k-2}{k-1}) \), let \( u := R(r)^{1/(k-1)} \) with \( u \in (0, 1) \). With this, we have

\[ r = S_k(u^{k-1}) = \frac{u - u^{k-1}}{1 - u^{k-1}}, \]

\[ 1 - r = \frac{1 - u}{1 - u^{k-1}} \]

and

\[ \frac{(1 - r)R(r)}{r + (1 - r)R(r)} = \frac{1 - u}{1 - u^{k-1}} + \frac{1 - u}{1 - u^{k-1}} u^{k-1} \]

\[ = \frac{(1 - u)u^{k-1}}{u - u^{k-1} + (1 - u)u^{k-1}} = \frac{(1 - u)u^{k-2}}{1 - u^{k-1}}. \]
Also
\[
\frac{dr}{du} = \frac{(1 - (k - 1)u^{k-2}) (1 - u^{k-1}) + (k - 1)u^{k-2} (u - u^{k-1})}{(1 - u^{k-1})^2}
= \frac{1 - (k - 1)u^{k-2} + (k - 2)u^{k-1}}{(1 - u^{k-1})^2}.
\]

Using (3.2) and (3.1), this is
\[
\frac{(1 - u)^2 \sum_{i=0}^{k-3} (i + 1)u^i}{(1 - u^{k-1}) (1 - u) \sum_{i=0}^{k-2} u^i} = \frac{(1 - u) \sum_{i=0}^{k-3} (i + 1)u^i}{(1 - u^{k-1}) \sum_{i=0}^{k-2} u^i}
\leq \frac{(1 - u)(k - 1) \left( \sum_{i=0}^{k-2} (k - 1)u^i \right)}{(1 - u^{k-1}) \sum_{i=0}^{k-2} u^i}
= \frac{(k - 1)(1 - u)}{1 - u^{k-1}}.
\]

Hence
\[
\int_0^{k-2} \left( \frac{(k - 1)(1 - r)R_k(r)}{r + (1 - r)R_k(r)} \right)^d dr
\leq \int_0^{k-2} (k - 1)^d \left( \frac{(1 - u)u^{k-2}}{1 - u^{k-1}} \right)^d dr
= \int_0^1 (k - 1)^d \left( \frac{(1 - u)u^{k-2}}{1 - u^{k-1}} \right)^d \frac{(k - 1)(1 - u)}{1 - u^{k-1}} du
= \int_0^1 (k - 1)^{d+1} \frac{(1 - u)^{d+1}u^{(k-2)d}}{(1 - u^{k-1})^{d+1}} du.
\]

Using (3.1), this is
\[
= \int_0^1 \frac{u^{(k-2)d}}{(1 - \sum_{i=0}^{k-2} u^i)^{d+1}} du.
\]

Using the arithmetic-geometric mean inequality, this is
\[
\leq \int_0^1 \frac{u^{(k-2)d}}{u^{\frac{(2-k)(d-1)}{d+1}}} \frac{1}{\sum_{i=1}^{d+1} u^{\frac{(k-2)(d-1)}{d+1}}} du
\leq \int_0^1 \frac{u^{(k-2)d}}{u^{\frac{(k-2)(d-1)}{d+1}}} du
= \int_0^1 u^{\frac{(k-2)(d-1)}{d+1}} du = \frac{2}{(k-2)(d-1) + 2}.
\]

By summing the two integrals, we obtain \( \Delta_k(d) \leq \frac{3}{(k-2)(d-1) + 2} \). \( \square \)
Chapter 4

General k-SAT

In this chapter, we provide the analysis of PPSZ for general $k$-SAT from the old version of [8], together with the improvements for 3-SAT of Iwama and Tamaki [5] and Rolf [13]. We also show the bound of Schöning [14]. We will consider 4-SAT in the next chapter, where we will also give better bounds for 3-SAT. In this chapter, let $F$ be a satisfiable $(\leq k)$-CNF.

4.1 Subcubes and Critical Clause Trees

To apply the concepts of the previous chapter to general $k$-SAT, we first show how to partition the space of all assignments into so called subcubes, such that in each subcube there is a unique satisfying assignment. We then compute the probability that a specific satisfying assignment $\alpha$ is returned given that the random assignment $\beta$ is in the corresponding subcube $B_\alpha$.

Let $V$ be a set of variables. Consider the set $\{0, 1\}^V$ of assignments on $V$. For $D \subseteq V$ and some assignment $\beta \in \{0, 1\}^V$, we define $B(D, \beta)$ to be the set of all assignments $\alpha$ with $\alpha|_D = \beta|_D$, i.e. the assignments agreeing with $\beta$ on $D$. $B(D, \beta)$ is called the subcube of $\{0, 1\}^V$ induced by $\beta|_D$. The variables in $D$ are called defining variables of $B(D, \beta)$. The variables in $V \setminus D$ are called nondefining variables of $B(D, \beta)$. The dimension of $B(D, \beta)$ is $|V \setminus D|$, the number of nondefining variables.

**Lemma 4.1 ([8])** Let $V$ be a set of variables, and let $A$ be a non-empty subset of $\{0, 1\}^V$. Then $\{0, 1\}^V$ can be partitioned into a family of disjoint subcubes $\{B_\alpha : \alpha \in A\}$ of $\{0, 1\}^V$ such that $\alpha \in B_\alpha$ for each $B_\alpha$.

**Proof** We use induction on $|V|$. If $|A|=1$, the statement is trivial. If $|V| = 0$, then also $|A| = 1$. 
Otherwise consider two distinct assignments $\alpha, \beta \in A$. Let $x \in V$ be such that $\alpha(x) \neq \beta(x)$. Let $V' := V \setminus \{x\}$. We use the induction hypothesis on the set of assignments of $A$ that assign 0 to $x$ (viewed as assignments on $V'$), and on the set of assignments that assign 1 to $x$ (again viewed as assignments on $V'$). The union of the two obtained families is then the family of subcubes we need for the statement. □

We now fix an arbitrary subcube partition for each satisfiable CNF $F$:

**Definition 4.2** For a satisfiable CNF $F$, we fix a partition of $\{0, 1\}^{V(F)}$ into subcubes $B_\alpha$ for $\alpha \in \text{sat}(F)$ according to Lemma 4.1. Let $D_\alpha$ be the set of defining variables of $B_\alpha$, and let $N_\alpha := V(F) \setminus D_\alpha$ be the set of nondefining variables.

We now consider PPSZ-runs, given $\beta$ is in some subcube $B_\alpha$. Let $F$ be a satisfiable CNF. Let $\beta \in \{0, 1\}^{V(F)}$ u.a.r. We define

$$p_{\text{PPSZ}}(F, s|B_\alpha) := \Pr(\text{PPSZ}(F, s, \beta) \in \text{sat}(F) | \beta \in B_\alpha)$$

and for $\alpha \in \text{sat}(F)$

$$p_{\text{PPSZ}}(\alpha, F, s|B_\alpha) := \Pr(\text{PPSZ}(F, s, \beta) = \alpha | \beta \in B_\alpha).$$

We have now

$$p_{\text{PPSZ}}(F, s) = \sum_{\alpha \in \text{sat}(F)} p_{\text{PPSZ}}(F, s|B_\alpha) \Pr(\beta \in B_\alpha)$$

$$\geq \sum_{\alpha \in \text{sat}(F)} p_{\text{PPSZ}}(\alpha, F, s|B_\alpha) \Pr(\beta \in B_\alpha).$$

This implies

$$p_{\text{PPSZ}}(F, s) \geq \min_{\alpha \in \text{sat}(F)} p_{\text{PPSZ}}(\alpha, F, s|B_\alpha). \quad (4.1)$$

That is, we need to bound for any fixed $\alpha \in \text{sat}(F)$ the probability that PPSZ returns $\alpha$ given $\beta \in B_\alpha$. We define

$$\text{Forced}(F, \pi, \alpha, s|B_\alpha) := \text{Forced}(F, \pi, \alpha, s) \cap N_\alpha,$$

the forced nondefining variables. Remember that we denote by $\pi_U$ the uniform random placement on $V(F)$. Analogous to Lemma 3.8, we have now

$$p_{\text{PPSZ}}(F, \alpha, s|B_\alpha) = E\left[2^{-|N_\alpha|+\text{Forced}(F,\pi_U,\alpha,s|B_\alpha)}\right]. \quad (4.2)$$

We could now apply Jensen’s inequality as in the unique case and bound $E[|\text{Forced}(F, \pi_U, \alpha, s|B_\alpha)|]$. However, we cannot bound the probability that a variable $x$ is forced as before. When we want to build a critical clause tree, it might now happen that the assignment $\alpha \oplus L$ satisfies $F$ and we cannot extend the tree. However, this only happens when there is a defining variable (w.r.t. $B_\alpha$) in $L$. This leads to the following lemma similar to Lemma 3.15. We extend the tree only on nodes labeled with nondefining variables.
4.1. Subcubes and Critical Clause Trees

Lemma 4.3 ([8]) Let $F$ be a satisfiable ($\leq k$)-CNF and $\alpha$ a satisfying assignment of $F$ with $B_\alpha$ as before. If $s \geq k^d$ then for every variable $x \in V(F)$ there exists a critical clause tree for $(x, \text{RESOLVE}(F, s), \alpha)$ of maximum degree $k - 1$, such that (i) any node labeled by a defining variable of $B_\alpha$ is a leaf, (ii) any leaf not labeled by a defining variable of $B_\alpha$ is at depth $d$.

Proof Analogous to the proof of Lemma 3.15; we only extend the tree for nodes that are unlabeled or labeled by a nondefining variable. By definition of $B_\alpha$, for a nonempty set $W \subseteq N_\alpha$ we have $\alpha \oplus W \in B_\alpha$ and thus $\alpha \oplus W \not\in \text{sat}(F)$. Therefore we can build the tree if the path to the root consists only of nondefining variables in the same way as in Lemma 3.15. The proof that the required critical clauses exist is identical. □

We see that it is possible that a critical clause tree consists just of the root and $k - 1$ children, all labeled with a defining variable. To compensate for that, we want to make defining variables more likely to occur at the beginning. We do this by looking only at some measurable subset $\Gamma$ of all placements; that subset should favor placements where defining variables occur at the beginning. We have

$$p_{PPSZ}(F, \alpha, s | B_\alpha) \geq E \left[ 2^{-|N_\alpha| + \text{Forced}(F, \pi_U, \alpha, s | B_\alpha)} \mid \pi_U \in \Gamma \right] \text{Pr}[\pi_U \in \Gamma].$$

Now we apply Jensen’s inequality, and we get

$$p_{PPSZ}(F, \alpha, s | B_\alpha) \geq 2^{-|N_\alpha| + E[\text{Forced}(F, \pi_U, \alpha, s | B_\alpha)]} \text{Pr}[\pi_U \in \Gamma].$$

As an example, we set $\Gamma$ to contain the placements $\pi$ where for $x \in D_\alpha$, we have $\pi \in \left[0, \frac{k - 2}{k - 1}\right]$, i.e. the defining variables must all have place at most $\frac{k - 2}{k - 1}$. With this, $\text{Pr}[\pi_U \in \Gamma] = \left(\frac{k - 2}{k - 1}\right)^{|D_\alpha|}$. One can also show that

$$E[\text{Forced}(F, \pi_U, \alpha, s | B_\alpha)] \mid \pi_U \in \Gamma] \geq R_k |N_\alpha|,$$

like in the unique case. Hence

$$p_{PPSZ}(F, \alpha, s | B_\alpha) \geq 2^{-(1 - R_k) |N_\alpha| + \log\left(\frac{k - 2}{k - 1}\right) |D_\alpha|}.$$ 

For $k \geq 5$, one can now show that $(1 - R_k) \geq - \log\left(\frac{k - 2}{k - 1}\right)$, implying that for $k \geq 5$ we have

$$p_{PPSZ}(F, \alpha, s | B_\alpha) \geq 2^{-(1 - R_k)n}$$

and unique $k$-SAT is the worst case. This approach corresponds essentially to the new version of [8]; we will consider $k \geq 5$ formally later in this chapter. In the old version of [8] a different and much more complicated approach was chosen, which gives a better bound for 3-SAT and is also used by Rolf [13].
4. General k-SAT

4.2 Nice Distribution Functions

We want the place of a defining variable to be chosen according to a more complicated distribution. For this, we define nice distribution functions that will induce such a distribution.

Definition 4.4 A function \( H : [0, 1] \rightarrow [0, 1] \) is called a nice distribution function if \( H \) is non-decreasing, uniformly continuous \(^1\), \( H(0) = 0 \), \( H(1) = 1 \), \( H \) is differentiable except for finitely many points and for \( r \in [0, 1] \), \( H(r) \geq r \).

We added the condition \( H(r) \geq r \) to what was required in [8]. This will be used in the next chapter. With a nice distribution function, we define the nice random placement \( \pi_H \). Remember that a nice random placement means that \( \pi(x) \) is chosen independently for every \( x \in V(F) \); nice random placements are the placements we can easily analyze.

Definition 4.5 Let \( H \) be a nice distribution function. By \( \pi_H \), we define the nice random placement on \( V(F) \) as follows: For \( x \in N_\alpha \), \( \pi(x) \) is chosen u.a.r. from \([0, 1]\). For \( x \in D_\alpha \), \( \pi(x) \) is chosen s.t. \( \Pr(\pi(x) \leq r) = H(r) \), i.e. with \( H(r) \) as distribution function.

The condition \( H(r) \geq r \) means that defining variables can only move to the beginning. Now we want to find some set \( \Gamma \) as before such that \( \pi_H \) and \( \pi_U \) given \( \pi_U \in \Gamma \) are distributed similarly; we will see later in which sense exactly. The proof that such a \( \Gamma \) exists is very technical. We have to bound two probabilities; the probability that \( \pi_U \in \Gamma \) and the probability that a variable is forced given \( \pi_U \in \Gamma \). We will show

\[
\Pr[\pi_U \in \Gamma] \geq 2^{-\beta_H + o(1)}|D_\alpha|, \tag{4.3}
\]

with

\[
\beta_H := \int_0^1 h(r) \log(h(r)) \, dr,
\]

where \( h(r) \) is the derivative of \( H(r) \). The negative of \( \beta_H \) is known as the differential entropy. One can check that this definition is consistent with the example we have given at the end of the previous section. The corresponding distribution function is \( H(r) = \max\left\{ \frac{k-1}{k-2}r, 1 \right\} \) with \( \beta_H = \log\left( \frac{k-1}{k-2} \right) \). We also show that

\[
E[|\text{Forced}(F, \pi_U, s|B_\alpha)| \mid \pi_U \in \Gamma] \geq (\gamma_H - o(1))|N_\alpha|, \tag{4.4}
\]

where

\[
\gamma_H := \int_0^1 \min\{H(r)^{k-1}, R_k(r)\} \, dr.
\]

These two equations imply the following theorem:

\(^1\) \( H \) is called uniformly continuous if for every real number \( \varepsilon > 0 \) there exists \( \delta > 0 \) such that for every \( s, t \in [0, 1] \) with \( |s - t| \leq \varepsilon \), we have \( |H(s) - H(t)| \leq \delta \).
4.3 Proof of (4.3) and (4.4)

In this technical section, we prove (4.3) and (4.4). If $|D_\alpha|$ is small, we are essentially in the unique case. Hence we can assume $|D_\alpha| \geq \sqrt{n(F)}$.

**Lemma 4.7** Let $F$ be a satisfiable $(\leq k)$-CNF. If $|D_\alpha| \leq n(F)/\log(n(F))$, we have

$$p_{PPSZ}(F, \alpha, s(n(F))) \geq 2^{-(1-R_k)n(F) - o(n(F))}.$$ 

**Proof** The probability that the variables in $D_\alpha$ come before the variables in $N_\alpha$ is

$$\frac{1}{n(F)} \geq \frac{1}{n(F)/\log(n(F))} \geq \frac{n(F)/\log(n(F))}{en(F)} \geq 2^{(\log(n(F)) - \log \log(n(F)) - \log(e))n(F)/\log(n(F))} \geq 2^{o(n(F))}.$$ 

If we assume that the variables in $D_\alpha$ come first, we know that there is only one satisfying assignment left. Hence we can use Theorem 3.30 together with $|D_\alpha|$ applications of Observation 3.2 with Lemma 2.3 to obtain the statement (the latter two lemmas correspond to the fact that we can restart PPSZ after the first $|D_\alpha|$ steps). 

In the following we fix $\alpha \in \text{sat}(F)$ and let $D := D_\alpha$ and $N := N_\alpha$. We now define the subset of placements $\Gamma$ we use.

**Definition 4.8 ([8])** Let $\Gamma_{H,\lambda}$ be the set placements $\pi$ with the property that for each $r > \lambda$, the number of defining variables mapped to $[0, r]$ is at least $H(r)|D|$.

First we show (4.3). $\beta_H$ can be characterized as a limit of an entropy:

**Lemma 4.9** Let $M \in \mathbb{N}$. For $i \in [M]$, let $g_i(M) := H(\frac{i}{M}) - H(\frac{i-1}{M})$. Then

$$\beta_H = \lim_{M \to \infty} \left( \log M + \sum_{i=1}^{M} g_i(M) \log g_i(M) \right).$$ 

Also

$$\beta_H \geq 0.$$ 

35
4. General k-SAT

Proof We have

\[
\log M + \sum_{i=1}^{M} g_i(M) \log g_i(M) = \frac{1}{M} \sum_{i=1}^{M} (Mg_i) \log(Mg_i).
\]

As \(Mg_i\) is between the minimum and maximum value of \(h\) in the interval \([m_i, m_{i+1}]\), this is a Riemann sum converging to \(\beta_H\) for \(M \to \infty\).

\(\beta_H \geq 0\) follows now, as the entropy of a random variable with \(M\) outcomes is bounded by \(\log M\). \(\square\)

The following Lemma implies (4.3):

Lemma 4.10 ([8]) If \(|D| \geq \sqrt{\frac{n}{N}}\), then

\[
\Pr(\pi \in \Gamma_{H,\lambda}) \geq 2^{\frac{|D|}{2}|(\beta_H + o(1))}
\]

where \(o(1)\) depends only on \(H\).

Proof Fix \(\varepsilon > 0\). Let \(M := \lfloor \sqrt{|D|} \rfloor\). Define \(m_i := \frac{i}{M}\) for \(0 \leq i \leq M\). We refer to the interval \([m_i, m_{i+1}]\) as bin \(i\). We define \(g_i := H(m_i) - H(m_{i-1})\).

We assume that \(|D|\) is large enough that

1. \(M > \frac{1}{\lambda}\).
2. \(H(\frac{2}{M}) < \varepsilon\).
3. \(|\beta_H - (\log M - \sum_{i=1}^{M} g_i \log g_i)| < \varepsilon\).

The second condition can be satisfied since \(H(0) = 0\) and \(H\) is continuous.

The third condition can be satisfied because of Lemma 4.9.

Define the \(M\)-profile of the placement \(\pi\) to be the sequence \(a_1(\pi), a_2(\pi), \ldots, a_M(\pi)\) where \(a_i(\pi)\) is the number of variables in \(D\) mapped to bin \(i - 1\).

A sufficient condition for \(\pi\) to be in \(\Gamma_{H,\lambda}\) is that for all \(j \in [M - 1]\), \(\sum_{i=1}^{j} a_i(\pi) \geq |D|H(m_{j+1})\). Define \(b_i := \lfloor |D|H(m_i) \rfloor - \lfloor |D|H(m_{i-1}) \rfloor\) for \(i \in [M]\). Let \(a_1 := b_1 + b_2, a_i := b_{i+1}\) for \(2 \leq i \leq M - 1\) and \(a_M = 0\). Note that \(a_i \geq 0, b_i \geq 0\), and \(\sum_{i=1}^{M} b_i = \sum_{i=1}^{M} a_i = |D|\). It follows that placement with \(M\)-profile \(a_1, \ldots, a_M\) is in \(\Gamma_{H,\lambda}\). The probability that \(\pi\) has profile \(a_1, \ldots, a_M\) is

\[
M^{-|D|} \frac{|D|!}{a_1! \ldots a_M!}.
\]

Using \(|D|! \geq \left(\frac{|D|}{e}\right)^{|D|}\), this is

\[
\geq \frac{1}{a_1! \ldots a_M!} \left(\frac{|D|}{eM}\right)^{|D|}.
\]
4.3. Proof of (4.3) and (4.4)

From Stirling’s approximation, we know there exists an \( x_0 \in \mathbb{N} \) s.t. for all \( x > x_0 \), we have \( x \leq \left( \frac{e}{x} \right)^{x+1} \). Hence we bound the \( a_i! \)-terms with \( a_i \leq x_0 \) from below by \( x_0! \), and we bound the other \( a_i! \)-terms by \( \left( \frac{e}{x} \right)^{x+1} \). Using this, we have

\[
\frac{1}{a_1! \cdots a_M!} \geq (x_0!)^{-M} a_1^{-a_1} \cdots a_M^{-a_M} \left( \frac{1}{e} \right)^{|D|-x_0} M \prod_{i,a_i > x_0} \left( \frac{e}{a_i} \right) .
\]

Using the arithmetico-geometric mean inequality, \( \prod_{i,a_i > x_0} \left( \frac{a_i}{b_i} \right) \leq \left( \frac{|D|}{M^2} \right)^M \). Putting things together, we obtain

\[
M^{-|D|} \frac{|D|!}{a_1! \cdots a_M!} \geq \left( \frac{x_0!}{e^{o(eM)}} \right)^{-M} a_1^{-a_1} \cdots a_M^{-a_M} \left( \frac{|D|}{M} \right)^{|D|} .
\]

Let \( p_i := \frac{a_i}{|D|} \) and \( q_i := \frac{b_i}{|D|} \). The previous expression can be written as

\[
2^{-|D|(\log M + \sum_{i=1}^M p_i \log p_i + o(1))}.
\]

We now show \( \left| \sum_{i=1}^M p_i \log p_i - \sum_{i=1}^M q_i \log q_i \right| \leq \epsilon + o(1) \). Assuming this, the statement follows by the third condition on \( M \) above, as \( \epsilon \) can be chosen arbitrarily small.

We have

\[
\left| \sum_{i=1}^M p_i \log p_i - \sum_{i=1}^M q_i \log q_i \right| \leq \left| \sum_{i=1}^M p_i \log p_i - \sum_{i=1}^M q_i \log q_i \right| + \left| \sum_{i=1}^M q_i \log q_i - \sum_{i=1}^M g_i \log g_i \right| .
\]

Now as \( p_1 = q_1 + q_2 \), \( p_i = q_{i+1} \) for \( 2 \leq i \leq M - 1 \) and \( p_M = 0 \), we have

\[
\left| \sum_{i=1}^M p_i \log p_i - \sum_{i=1}^M q_i \log q_i \right| \leq (q_1 + q_2) (\log(q_1 + q_2)) - q_1 \log q_1 - q_2 \log q_2 .
\]

With \( \gamma := q_1 / (q_1 + q_2) \), this is equal to

\[
(q_1 + q_2) | - \gamma \log \gamma - (1 - \gamma) \log(1 - \gamma) | 
\leq q_1 + q_2 \leq H(m_2) + \frac{1}{|D|} \leq \epsilon + \frac{1}{D} = \epsilon + o(1) .
\]

For the second part, we have

\[
\left| \sum q_i \log q_i - \sum g_i \log g_i \right| \leq \sum |q_i \log q_i - g_i \log g_i| .
\]
We now claim that for $0 < s \leq \frac{1}{e}$ and $0 < t \leq \frac{1}{e}$, we have $|s \log s - t \log t| \leq ||s - t|| \log |s - t||$. W.l.o.g., we assume $s \geq t$. Consider for $x \in [0, \frac{1}{e}]$ the function $f(x) := -x \log x$. As $f(x) \geq 0$, we have $f(x) = |x \log x|$. For $x \in (0, \frac{1}{e})$, we have $f'(x) = -\frac{1}{\ln 2} - \log(x) \geq 0$, hence $f$ is monotonically decreasing. Also, as $f''(x) = -\frac{1}{\ln(2)x} < 0$, $f$ is concave. For concave functions, the following holds:

**Proposition** If $f$ is a concave function, then for $s \geq t \geq 0$ we have

$$f(s) - f(t) \leq f(s - t) - f(0).$$

**Proof** By concavity, we have

$$f(t) \geq \frac{t}{s} f(s) + \left(1 - \frac{t}{s}\right) f(0),$$

$$f(s - t) \geq \frac{s - t}{s} f(s) + \left(1 - \frac{s - t}{s}\right) f(0).$$

Summing both inequalities gives

$$f(t) + f(s - t) \geq f(s) + f(0)$$

and hence

$$f(s) - f(t) \leq f(s - t) - f(0).$$

Now, as $f$ is decreasing, we now have

$$|s \log s - t \log t| = (-s \log s) - (-t \log t)$$

$$= f(s) - f(t) \leq f(s - t) - f(0) = -(s - t) \log(s - t) - 0$$

$$= ||s - t|| \log |s - t||,$$

proving the claim.

As $H$ is uniformly continuous, for $|D|$ large enough, we have for all $i$ that $q_i \leq \frac{1}{e}$ and $g_i \leq \frac{1}{e}$. Hence

$$|q_i \log q_i - g_i \log g_i| \leq ||q_i - g_i|| \log |q_i - g_i||.$$

As $|q_i - g_i| \leq \frac{1}{|D|}$, this is at most $\frac{\log |D|}{|D|}$. Hence at sum is at most $\frac{M \log |D|}{|D|} = o(1)$. □

Now we prove (4.4), stating

$$E[|\text{Forced}(F, \pi_U, \alpha, s|B_k)| \mid \pi_U \in \Gamma] \geq (\gamma_H - o(1))|N_k|.$$

We write $\pi_{H,\lambda}$ for $\pi_U$ given $\pi_U \in \Gamma_{H,\lambda}$:
We want to show that \( \pi \) and \( \pi_{H,\lambda} \) are similar. However, in general \( \pi_H \) and \( \pi_{H,\lambda} \) are quite different; the former can attain values the latter cannot. We can show the similarity only in a special case, leading to the following technical lemma:

**Lemma 4.13 ([8])** Let \( s := \log(n(F)) \), and let \( S \) be a subset of \( V(F) \) with \( |S| \leq s \). Let \( g \) be a function from \( 2^S \) to \( \{0,1\} \) with the property that for \( W' \subseteq W \), \( g(W') \leq g(W) \) (i.e. \( g \) is monotone increasing). Let \( r' \in [0,1] \) u.a.r. Let \( S(\pi, r) := \{ v \in S : \pi(v) < r \} \).

If \( |D| \geq \sqrt{n(F)} \), then

\[
E \left[ g(S(\pi_{H,\lambda}, r')) \right] \geq E \left[ g(S(\pi_H, r')) \right] - (\lambda + o(1)),
\]

where \( o(1) \) only depends on \( H \).

**Proof** Remember that \( \pi_U \) denotes the uniform random placement on \( V(F) \). First we consider given \( r \in [\lambda,1] \). Let \( w(r) := \lfloor |D| H(r) \rfloor \). Let \( D(\pi, r) := \{ v \in D : \pi(v) < r \} \). Observe that \( \pi \in \Gamma_{H,\lambda} \) implies that \( |D(\pi, r)| \geq w(r) \) for \( r \geq \lambda \). If we condition on \( |D(\pi_U, r)| = w \) for some \( w \geq w(r) \), then \( D(\pi_U, r) \) is uniformly distributed among all subsets of \( D \) of size \( w \). Conditioning further on \( \pi_U \in \Gamma_{H,\lambda} \) does not change the distribution of \( D(\pi_U, r) \). Let \( W(\pi, r) \) denote the event \( |D(\pi, r)| = w(r) \). We have

\[
E \left[ g(S(\pi_{H,\lambda}, r)) \right] = \sum_{w=w(r)}^{|D|} E \left[ g(S(\pi_{H,\lambda}, r)) \mid |D(\pi_{H,\lambda}, r)| = w \right] \Pr \left( |D(\pi_{H,\lambda}, r)| = w \right).
\]

By the previous observations and the definition of \( g \), it is easily seen that \( E \left[ g(S(\pi_{H,\lambda}, r)) \mid |D(r)| = w \right] \) is monotonically increasing in \( w \). Hence the previous expression is at least

\[
E \left[ g(S(\pi_{H,\lambda}, r)) \mid W(\pi_{H,\lambda}, r) \right].
\]

Let \( S' := D \cap S \) and let \( S'(\pi, r) := S \cap D(\pi, r) = S(\pi, r) \cap D \).

Let \( \rho(r) := \min \left\{ \frac{x^2}{|D| H(x-1)} : x \in (0,1) \right\} \), and \( \rho(0) = \rho(1) = 0 \). We now want to show for fixed \( r \in [\lambda,1] \) that

\[
E \left[ g(S(\pi_{H,\lambda}, r)) \right] \geq E \left[ g(S(\pi_H, r)) \right] - \rho(H(r)).
\]

It is easily seen that \( S'(\pi_{H,\lambda}, r) \) and \( S'(\pi_H, r) \) are independent of \( S(\pi_{H,\lambda}, r) \setminus D \) and \( S(\pi_H, r) \setminus D \), respectively. Also, \( S(\pi_{H,\lambda}, r) \setminus D \) and \( S(\pi_H, r) \setminus D \) have
the same distribution, where each element is chosen independently with probability \( r \). Hence if \( H(r) = 1 \) or \( H(r) = 0 \), then \( S(\pi_{H, r}) \) and \( S(\pi_{H', r}) \) have the same distribution, as in this case \( S'(\pi_{H, r}) = S'(\pi_{H', r}) = S' \) or \( S'(\pi_{H, r}) = S'(\pi_{H', r}) = \emptyset \), respectively. Therefore we can assume \( H(r) \in (0, 1) \), and we have

\[
\frac{E[g(S(\pi_{H, r}))]}{E[g(S(\pi_{H', r}))]} \geq \frac{E[g(S(\pi_{H, r})) | W(\pi_{H, r})]}{E[g(S(\pi_{H', r})) | W(\pi_{H, r})]}
\]

Denote by \( A(\pi, C) \) the event \( S'(\pi, r) = C \). The previous expression is equal to

\[
\sum_{C \subseteq S'} E[g(S(\pi_{H, r})) | W(\pi_{H, r})] \Pr(A(\pi_{H, r}, C) | W(\pi_{H, r})) \Pr(A(\pi_{H, r}, C))
\]

If \( a_1, \ldots, a_t \) and \( b_1, \ldots, b_t \) are nonegative reals, then \( \left( \sum_i b_i \right) / \left( \sum_i a_i \right) \geq \min_i b_i / a_i \). Hence the previous expression is

\[
\geq \min_{C \subseteq S'} \frac{E[g(S(\pi_{H, r})) | W(\pi_{H, r})] \Pr(A(\pi_{H, r}, C) | W(\pi_{H, r}))}{E[g(S(\pi_{H', r})) | W(\pi_{H, r})] \Pr(A(\pi_{H, r}, C))}
\]

By the previous observations, the two expectations are equal, and we get

\[
\frac{E[g(S(\pi_{H, r}), r))]}{E[g(S(\pi_{H', r}))]} \geq \min_{C \subseteq S'} \frac{\Pr(A(\pi_{H, r}, C) | W(\pi_{H, r}))}{\Pr(A(\pi_{H, r}, C))}
\]

From the observations at the beginning of the proof it follows that the distribution of \( S'(\pi_{H, r}) \) conditioned on \( W(\pi_{H, r}) \) is induced by choosing a subset \( C' \) of \( D(\pi, r) \) u.a.r. from the sets of size \( \omega(r) \) and intersecting it with \( S' \). We have for \( C \subseteq S' \) that

\[
\Pr(A(\pi_{H, r}, C) | W(\pi_{H, r})) = \frac{\prod_{i=0}^{\lfloor |C| \rfloor (|D| - i)} \prod_{i=0}^{\lfloor |S' - |C| \rfloor (|D| - i - \omega(r) - i)}}{\prod_{i=0}^{\lfloor |S' - |C| \rfloor (|D| - i)}}
\]

\[
\geq \left( \frac{H(r)|D| - |C|}{|D|} \right)^{|C|} \left( \frac{1 - H(r)}{|D|} \right)^{|C|} \left( \frac{|S'| - |C|}{|H(r)|D|} \right)^{|S' - |C|}
\]

\[
= H(r)^{|C|} (1 - H(r))^{|S' - |C|} \left( \frac{|S'|}{|H(r)|D|} \right)^{|C|} \left( \frac{1 - \omega(r)}{1 - H(r)} \right)^{|S' - |C|}
\]

\[
\geq \Pr(A(\pi_{H, r}, C)) \left( 1 - \min \left\{ \frac{|S'|^2}{|D|} \left( \frac{1}{H(r)} + \frac{1}{1 - H(r)} \right), 1 \right\} \right)
\]

\[
\geq \Pr(A(\pi_{H, r}, C)) \left( 1 - \rho(H(r)) \right).
\]
This proves that for \( r \in [\lambda, 1] \), we have
\[
E \left[ g(S(\pi_{H, r})) \right] \geq E \left[ g(S(\pi_{H, r})) (1 - \rho(H(r))) \right] - \rho(H(r)).
\]

We now apply Lemma 4.13 for each \( r \in \), we have
\[
E \left[ g(S(\pi_{H, r})) \right] \geq E \left[ g(S(\pi_{H, r})) \right] (1 - \rho(H(r))) - \rho(H(r)).
\]

To conclude the proof, we have show
\[
E \left[ g(S(\pi_{H, r})) \right] \leq \lambda - \int_0^1 \rho(H(r)) dr.
\]

Now, we have
\[
\int_0^1 \rho(H(r)) dr = \int_{r_0}^{r_1} \rho(H(r)) dr \leq \int_{r_0}^{r_0 + \epsilon} \rho(H(r)) dr + \int_{r_0 + \epsilon}^{r_1 - \epsilon} \rho(H(r)) dr + \int_{r_1 - \epsilon}^{r_1} dr
\]
\[
= 2\epsilon + \int_{r_0 + \epsilon}^{r_1 - \epsilon} \frac{s^2}{|D|} \left( \frac{1}{H(r)} \frac{1}{1 - H(r)} \right) dr
\]
\[
\leq 2\epsilon + \frac{s^2}{|D|} \left( \frac{1}{H(r_0 + \epsilon)} \frac{1}{1 - H(r_1 - \epsilon)} \right).
\]

Since \( H \) is fixed and \( s = \log(n(F)) \), if \( n(F) \) is sufficiently large then, as \( |D| \geq \sqrt{n(F)} \), the second term is at most \( \epsilon \). As \( \epsilon \) can be chosen arbitrarily small, this is \( o(1) \).

Now, we have
\[
E[\text{Forced}(F, \pi_U, a, s|B_a) \mid \pi_U \in \Gamma_{H, \lambda}]
\]
\[
= E[\text{Forced}(F, \pi_{H, \lambda}, a, s|B_a)].
\]

For nondefining variables \( x \), let \( T_x \) be a tree obtained from Lemma 4.3 (with \( d := \log_n(s) \)). Using a similar argument as in Lemma 3.16, we have
\[
E[\text{Forced}(F, \pi_{H, \lambda}, a, s|B_a)] \geq \sum_{x \in N} Q(T_x, \pi_{H, \lambda}).
\]

We now apply Lemma 4.13 for each \( x \in \): We let \( S \) be the set of variables occurring as labels in \( T_x \), and let \( g(S') \) be the indicator function that \( S' \) is a cut in \( T_x \). With this, we have \( |S| \leq \log n \) and the we get
\[
\sum_{x \in N} Q(T_x, \pi_{H, \lambda}) \geq \sum_{x \in N} Q(T_x, \pi_{H}) - \lambda - o(1).
\]
As $\lambda$ can be chosen arbitrarily small, we have

$$E[|\text{Forced}(F, \pi_{H,\lambda}, \alpha, s|B_\alpha)|] \geq \sum_{x \in N} Q(T_x, \pi_H) - o(1). \quad (4.5)$$

We now have to bound $Q(T_x, \pi_H)$ for $x \in N$. Applying Lemma 3.18 to $\pi_H$, we obtain the following corollary:

**Corollary 4.14 ([8])** Consider the nice random placement $\pi_H$. Let $T$ be an admissible tree with at least two nodes. Let $r \in [0, 1]$. Let $T_1, \ldots, T_t$ be the (admissible) subtrees of $T$ rooted at those children of the root of $T$ that are labeled. Let $T_1, \ldots, T_s$ be those whose root is labeled by a variable in $D$. Let $x_i := L(\text{Root}(T_i))$. Then

$$Q(T, r) \geq H(r)^s \prod_{i=s+1}^t (r + (1 - r)Q(T_i, r)).$$

The following lemma from [8] has been generalized by [13]. The statement is similar to Lemma 3.22.

**Lemma 4.15 ([8],[13])** For $x \in N$, let $T_x$ be a tree obtained form Lemma 4.3. Then

$$Q(T_x, \pi_H) \geq \gamma_H - \varepsilon_k(d).$$

**Proof** Given the placement $\pi_H$, and given the critical clause tree $T$ for $x$ obtained from Lemma 4.3 where leaves labeled by variables in $N$ have depth $d$, we show by induction on $d$ that

$$Q(T, r) \geq \min\{H(r)^{k-1}, Q_k(d, r)\}.$$ 

By definition of $\gamma_H$ and by Lemma 3.26 the statement then follows.

For $d = 0$, $Q_k(0, r) = 0$ by definition and the statement is trivial. Assume the statement holds for $d - 1$. By Corollary 4.14, we have with $T_i$ defined accordingly,

$$Q(T, r) \geq H(r)^s \prod_{i=s+1}^t (r + (1 - r)Q(T_i, r)).$$

As in $T_i$ leaves labeled by variables in $N$ have depth $d - 1$, we can use the induction hypothesis, and we get (remember $f(t, r) := (r + (1 - r)t)^{k-1}$)

$$Q(T, r) \geq H(r)^sf(Q_k(d - 1, r), r)^{(t-s)/(k-1)} = H(r)^s Q_k(d, r)^{(t-s)/(k-1)}.$$

As $T$ has degree at most $k - 1$, $t \leq k - 1$, and as $H(r) \leq 1$ and $Q_k(d, r) \leq 1$, $Q(T, r) \geq \min\{H(r)^{k-1}, Q_k(d, r)\}$. □

By the choice of $s$ and by setting $d = \lfloor \log_k s \rfloor$, this now implies (4.4).
4.4 Bound for \( k \geq 5 \)

In this section we show that for \( k \geq 5 \), we get the same running time for general \( k \)-SAT as for unique \( k \)-SAT. This was proved in [8]. The proofs are similar to the new version of [8]. In this section, we use \( H(r) := \min \left\{ \frac{k-1}{k-2} r, 1 \right\} \).

This choice of \( H \) has been pointed out by [13]. It is easily seen that this \( H \) is a nice distribution function. The following lemma holds also for \( k = 3 \) and \( k = 4 \):

**Lemma 4.16 ([8])** For \( r \in [0,1] \), we have

\[ H(r)^{k-1} \geq R_k(r). \]

This implies that \( \gamma_H = R_k \). See Figure 4.1 for a plot with \( k = 5 \).

**Proof** For \( r \in \left[ \frac{k-2}{k-1}, 1 \right] \), this is trivial, as there \( H(r)^{k-1} = 1 \). For \( r \in [0, \frac{k-2}{k-1}] \), \( S_k \) is the inverse of \( R_k \), hence the statement is equivalent to

\[ S_k \left( \left( \frac{k-1}{k-2} r \right)^{k-1} \right) \geq r. \]
4. General k-SAT

Setting $s := \frac{k-1}{k-2}r$, we need to show for $s \in [0,1]$ that

$$S_k \left( s^{k-1} \right) \geq \frac{k-2}{k-1}s.$$  

We have by Lemma 3.31

$$S_k \left( s^{k-1} \right) = 1 - \frac{1}{\sum_{i=0}^{k-2} s^i}$$

$$= \frac{\sum_{i=1}^{k-2} s^i}{\sum_{i=0}^{k-2} s^i} = \left( \frac{\sum_{i=0}^{k-3} s^i}{\sum_{i=0}^{k-2} s^i} \right) s$$

$$= \left( 1 - \frac{s^{k-2}}{\sum_{i=0}^{k-2} s^i} \right) s.$$  

For $i \in [k-2]$, we have $s^i \geq s^{k-2}$, hence the last expression is

$$\geq \left( 1 - \frac{1}{k-1} \right) s = \frac{k-2}{k-1}s.$$  

The important fact for $k = 5$ is that there is a $H$ with $\gamma_H = R_k$ and $\beta_H \leq 1 - \gamma_H$. The following lemma states the latter property:

**Lemma 4.17** For $k \geq 5$, we have $\beta_H \leq 1 - \gamma_H$.

**Proof** Note that this can be easily seen by numerical computation for $k = 5$ with the observation that $\beta_H$ and $\gamma_H$ decrease with increasing $k$. We give a more formal proof adapted from the new version of [8].

By the previous lemma, we have $\gamma_H = R_k$. Hence by Lemma 3.29, $1 - \gamma_H = S_k$. Using Lemma 3.32 and Lemma 3.29, we have

$$S_k = 1 - \frac{1}{k-1} \sum_{j=1}^{\infty} \frac{1}{j \left( j + \frac{1}{k-1} \right)}$$

$$\geq 1 - \frac{1}{k-1} \sum_{j=1}^{\infty} \frac{1}{j^2}.$$  

$$= 1 - \frac{\pi^2}{k-1} \cdot \frac{6}{6}.$$  

For $k \geq 5$, this is

$$\geq 1 - \frac{110}{4} = \frac{5}{12}.$$  

It is easily seen that $\beta_H = \log \left( \frac{k-1}{k-2} \right)$. Hence for $k \geq 5$, we have

$$\beta_H \geq \log \left( \frac{4}{3} \right).$$  

By numerical computations, we have now $\frac{5}{12} > 0.416 > \log \left( \frac{4}{3} \right).$  

□
Now we immediately get the following theorem:

**Theorem 4.18** Let $k \geq 5$. If $F$ is a satisfiable $(\leq k)$-CNF, we have

$$p_{PPSZ}(F, \alpha, s(n(F))) \geq 2^{-Sn(F) - o(n(F))}.$$  

There is a randomized algorithm for $k$-SAT with running time $O\left(2^{Sn(F) + o(n(F))}\right)$.

**Proof** The statements follow by the Theorem 4.6 using the two lemmas proved in this section. □

## 4.5 Schöning’s Algorithm

For 3-SAT, having many defining variables is bad, as $1 - \gamma_H$ (the fraction of guessed variables, roughly 0.4) is small and $\beta_H$ (the “cost” of a defining variable, roughly 0.9) is rather large. Iwama and Tamaki [5] discovered that Schöning’s algorithm [14] is very good in these cases. This leads to a combined algorithm $\text{COMB}$. We define $\text{SCHÖNING}(F, \beta)$ and $\text{COMB}(F, s)$ in pseudocode. It is important that the same initial assignment is used for both $\text{PPSZ}$ and $\text{SCHÖNING}$; we decide which algorithm to analyze depending on the fraction of defining variables of the subcube the assignment lies in.

### Algorithm 6 $\text{SCHÖNING}(\text{CNF } F, \text{ assignment } \beta)$

```plaintext
for 3n(F) steps do
    if $\beta$ does not satisfy $F$ then
        Select an arbitrary $C \in F$ not satisfied by $\beta$
        Select a variable $x$ u.a.r. from $V(C)$ and flip $x$ in $\beta$
    end if
end for
return $\beta$
```

### Algorithm 7 $\text{COMB}(\text{CNF } F, \text{ integer } s)$

```plaintext
For all $x \in V(F)$, choose $\beta(x)$ u.a.r. from $\{0, 1\}$
$\alpha \leftarrow \text{PPSZ}(F, s, \beta)$
if $\alpha \not\in \text{sat}(F)$ then
    $\alpha \leftarrow \text{SCHÖNING}(F, \beta)$
end if
return $\alpha$
```

### Definition 4.19 Let $\sigma_k := \log \frac{k}{2(k-1)}$.

Let $\sigma := \sigma_3 = \log \frac{3}{4}$. 

45
The following theorem about the Schöning’s algorithm [14] is used in the bound of Iwama and Tamaki [5] and Rolf [13].

**Theorem 4.20** ([14], [5]) Let $F$ be a satisfiable $(\leq k)$-CNF. Let $B$ be a subcube of $\{0, 1\}^{V(F)}$ with defining variables $N$ s.t. $B \cap \text{sat}(F) \neq \emptyset$. Then for $\beta$ chosen u.a.r. from $B$, we have

$$\Pr(\text{Schoening}(F, \beta) \in \text{sat}(F) \mid \beta \in B) \geq 2^{-\alpha_l|N| - o(n(F))}.$$ 

Note that the dimension of the subcube is $|N|$. We only need this theorem for 3-SAT; we were not able to use it for 4-SAT, as we will see in the next chapter.

**Proof** We adapt the proof from [15]. Fix a satisfying assignment $\alpha \in B$. Consider step $i$ of SCHOENING. If $\beta$ does not satisfy $F$, then the chosen clause $C$ contains at least one literal $x_i$ satisfied by $\alpha$. Let $A_i$ be the event that $x_i$ is chosen. We have $\Pr(A_i) = \frac{1}{k}$. If $\beta$ does satisfy $F$, let $A_i$ be an event that occurs independently with probability $\frac{1}{k}$. Assume that $\beta$ has Hamming distance $j$ to $\alpha$. Let $r := \frac{1}{k-2}$. Consider the first $(1+2r)j$ steps in SCHOENING($F, \beta$). It is easily seen if for exactly exactly $(1+r)j$ steps $i$ the event $A_i$ occurs, then SCHOENING($F, \beta$) returns a satisfying assignment of $F$. This happens with probability

$$q_j := \left(\frac{(1+2r)j}{(1+r)j}\right) \left(\frac{k-1}{k}\right)^{rj} \left(\frac{1}{k}\right)^{(1+r)j}.$$ 

By Stirling’s approximation, we have

$$\left(\frac{(1+2r)j}{(1+r)j}\right) = \frac{((1+2r)j)!}{(rj)! ((1+r)j)!} = \Omega\left(j^{-\frac{1}{2}} \left(\frac{(1+2r)j}{e}\right)^{(1+2r)j} \left(\frac{e}{rj}\right)^{rj} \left(\frac{e}{1+r}\right)^{(1+r)j}\right)$$

$$= \Omega\left(j^{-\frac{1}{2}} \left(\frac{(1+2r)^{1+2r}}{r(1+r)^{1+r}}\right)^j\right)$$

$$= \Omega\left(j^{-\frac{1}{2}} \left(\frac{1+2r}{r}\right)^{rj} \left(\frac{1+2r}{1+r}\right)^{(1+r)j}\right).$$

We have $1+2r = \frac{k}{k-2}$ and $1+r = \frac{k-1}{k-2}$. Hence

$$\frac{1+2r}{1+r} = \frac{k}{k-1}$$
and
\[
\frac{1 + 2r}{r} = k.
\]

Hence
\[
\left(\frac{(1 + 2r)^j}{(1 + r)^j}\right) = \Omega\left(j^{-\frac{1}{2}} k^j \left(\frac{k}{k-1}\right)^{(1+r)j}\right).
\]

Hence
\[
q_j \geq \Omega\left(j^{-\frac{1}{2}} k^j \left(\frac{k}{k-1}\right)^{(1+r)j} \frac{1}{k} \left(1 - \frac{1}{k}\right)^{(1+r)j}\right) = \Omega\left(j^{-\frac{1}{2}} (k-1)^j \frac{1}{k-1} \left(\frac{1}{k}\right)^{(1+r)j}\right).
\]

Now consider \(\beta\) chosen u.a.r. from \(B\), and a satisfying assignment \(\alpha \in B\).
The distance between \(\beta\) and \(\alpha\) is binomially distributed with parameter \(|N|\) and \(\frac{1}{2}\). Hence the probability that \(\text{SCHÖNING}\,(F, \beta)\) returns a satisfying assignment is at least
\[
2^{-|N|} \sum_{j=0}^{|N|} q_j = 2^{-|N|} \sum_{j=0}^{|N|} \Omega\left(j^{-\frac{1}{2}} \left(\frac{n}{j}\right) \left(\frac{1}{k-1}\right)^j\right) \geq \Omega\left(n(F)^{-\frac{1}{2}} \left(\frac{1}{k-1}\right)^{|N|}\right) = \Omega\left(n(F)^{-\frac{1}{2}} \left(\frac{k}{2(k-1)}\right)^{|N|}\right).
\]

The statement now follows.

Given \(H\), we can state which fraction of defining variables is the worst case.
This occurs when the probabilities of PPSZ and \(\text{SCHÖNING}\) are equal.

**Theorem 4.21 ([13])** Let \(H\) be a nice distribution with \(\beta_H \geq 1 - \gamma_H\), \(\sigma_k > 1 - \gamma_H\). Then for
\[
\delta_H = \frac{\sigma_k + \gamma_H - 1}{\sigma_k + \gamma_H + \beta_H - 1},
\]
we have
\[
\Pr\left(\text{COMB}(F, s) \in \text{sat}(F)\right) \geq 2^{-\sigma_k(1-\delta_H)n(F) - o(n(F))}.
\]

Note that we were only able to use this theorem for \(k = 3\). We do not know if for a more refined choice of \(H\) it can be used for \(k = 4\).
4. General k-SAT

**Proof** It is easily seen that if the required conditions hold, then $\delta_H$ is well-defined with $0 < \delta_H \leq 1$ (note that we have shown $\beta_H \geq 0$ in Lemma 4.9).

Consider $\alpha \in \text{sat}(F)$. We have by Theorem 4.6

$$p_{\text{PPSZ}}(F, \alpha, s(n(F)) | B_a) \geq 2^{-\beta_H|D_a|+(1-\gamma_H)|N_a|-o(n(F))}$$

and by Theorem 4.20

$$\Pr(\text{Schoening}(F, \beta) \in \text{sat}(F) \mid \beta \in B) \geq 2^{-\alpha_k|N|-o(n(F))}.$$ 

Solving the following equation for $\delta_H$ gives the fraction of defining variables for which PPSZ and Schoening have the same probability:

$$\beta_H\delta_H + (1 - \gamma_H)(1 - \delta_H) = \sigma_k(1 - \delta_H).$$

We get

$$\beta_H\delta_H + 1 - \delta_H - \gamma_H + \gamma_H\delta_H = \sigma_k - \sigma_k\delta_H$$

and

$$\beta_H\delta_H - \delta_H + \gamma_H\delta_H + \sigma_k\delta_H = -1 + \gamma_H + \sigma_k$$

and hence

$$\delta_H = \frac{\sigma_k + \gamma_H - 1}{\sigma_k + \gamma_H + \beta_H - 1}.$$ 

For $|D_a| \leq \delta_H n(F)$, as we required $\beta_H \geq 1 - \gamma_H$, the probability of PPSZ increases, and trivially for $|D_a| \geq \delta_H n(F)$, the probability of Schoening increases.

Hence for $\alpha \in \text{sat}(F)$, we have

$$\Pr(\text{Comb}(F, s) \in \text{sat}(F) \mid \beta \in B_a) \geq 2^{-\alpha_k(1-\delta_H)n(F)-o(n(F))}.$$ 

As this holds for all $\alpha \in \text{sat}(F)$, this implies the claim, by an argument similar to (4.1). $\square$

### 4.6 Optimized Distribution Functions for 3-SAT

In this section, we present the choice of $H$ from Rolf [13], and evaluate it with Theorem 4.21. This corresponds directly to Section 6 in [13]. See Figure 4.2 for a plot of $H$.

Let now $k = 3$. Let $\theta \in [0.5, 1]$ be a parameter. With some appropriate parameters $a$ and $b > 1$, we define $H(r)$ as follows:

$$H(r) := \begin{cases} 
    r/\theta & \text{if } r \in [0, 1 - \theta) \\
    1 - (-(a \ln(r))^b) & \text{if } r \in [1 - \theta, 1]
\end{cases}$$
4.6. Optimized Distribution Functions for 3-SAT

Figure 4.2: $H(r)$ for 3-SAT from [13]

Let $h$ be the derivative of $H$. We have

\[ h(r) = \begin{cases} 
\frac{1}{\theta} & \text{if } r \in [0, 1 - \theta) \\
-b \frac{(-a \ln(r))^b}{r \ln(r)} & \text{if } r \in [1 - \theta, 1)
\end{cases} \]

To determine the optimal values for $a$ and $b$, we want the following constraints to be satisfied.

\[ H(1 - \theta) = R_3(1 - \theta)^{1/2} \]

(as $\theta \geq 1/2$, this right-hand side is equal to $\frac{1 - \theta}{\theta}^2$) and

\[ h(1 - \theta) = \frac{1}{\theta}. \]

If these constraints are satisfied, it is easy to check that $H(r)$ is a nice distribution function and that is differentiable everywhere.

If we set

\[ a := -\left( \frac{2\theta - 1}{\theta} \right)^{2\theta - 1} (\ln(1 - \theta))^{-1} \]

and

\[ b := \frac{\ln(1 - \theta)(\theta - 1)}{2\theta - 1}, \]
both constraints are satisfied. This can be easily seen by first checking

\[ (-a \ln(1 - \theta))^b = \frac{2\theta - 1}{\theta}, \]

and then inserting this into the constraints.

We now need to compute \( \beta_H \) and \( \gamma_H \) and insert these values into Theorem 4.21. Obviously, for \( r \in [0, 1 - \theta] \), we have \( R_3(r)^{1/2} \leq H(r) \). For \( r \in [1 - \theta, 1] \), we have \( H(r) \leq R_3(r)^{1/2} \). This is because for \( r \in [1/2, 1] \), we have \( R_3(r)^{1/2} = 1 \) and \( H(r) \leq 1 \). For \( r \in [1 - \theta, 1/2) \), it is easily checked that for \( b > 1 \), the derivative of \( H(r) \) is decreasing (as \( H(r) \) is a concave function), and the derivative of \( R_3(r)^{1/2} \) is increasing (as \( R(r)^{1/2} \) is a convex function).

We now have to compute \( \beta_H \) and \( \gamma_H \). We have

\[ \beta_H = \int_0^1 h(r) \log h(r) dr = \beta_1 + \beta_2, \]

where

\[ \beta_1 := \int_0^{1-\theta} h(r) \log h(r) dr, \]

\[ \beta_2 := \int_{1-\theta}^1 h(r) \log h(r) dr. \]

We have trivially

\[ \beta_1 = \int_0^{1-\theta} \frac{1}{\theta} \log \frac{1}{\theta} dr = \frac{1 - \theta}{\theta} \log \frac{1}{\theta}. \]

We claim that for \( r \in [1 - \theta, 1) \), the antiderivative of \( h(r) \log h(r) \) is

\[ \beta_2(r) := \frac{-(-a \ln r)^b \left( b \ln r - b^2 + 1 + (b + b^2) \ln \left( -b \frac{(-a \ln r)^b}{r \ln r} \right) \right)}{(b + b^2) \ln 2} + C. \]

We check this by derivation. Define

\[ s(r) := -(-a \ln r)^b. \]

Note that \( s(r) = H(r) - 1 \). We have

\[ s'(r) = h(r) = -b \frac{(-a \ln r)^b}{r \ln r} = b \frac{s(r)}{r \ln r}, \]

and

\[ s''(r) = b \frac{s'(r)(r \ln r) - s(r)(\ln(r) + 1)}{(r \ln r)^2}. \]
\[ \frac{b s'(r)}{r \ln r} - \frac{s'(r)}{r \ln r} (\ln r + 1) = \frac{s'(r)}{r \ln r} (b - 1 - \ln r). \]

With this, we have
\[ \beta_2(r) = \frac{s(r) \left( b \ln r - b^2 + 1 + (b + b^2) \ln (s'(r)) \right)}{(b + b^2) \ln 2} + C. \]

For the derivative, we have
\[ (b + b^2) \ln(2) \cdot \beta'_2(r) = s'(r) \left( b \ln r - b^2 + 1 + (b + b^2) \ln (s'(r)) \right) + s(r) \left( \frac{b}{r} + (b + b^2) \frac{s''(r)}{s'(r)} \right) \]
\[ = s'(r) b \ln(r) - s'(r) b^2 + s'(r)(b + b^2) \ln (s'(r)) + s(r) \left( \frac{b}{r} + s(r)(b + b^2) \frac{s''(r)}{s'(r)} \right) \]
\[ = b \ln(r) s'(r) - b^2 s'(r) + s'(r) + (b + b^2) s'(r) \ln (s'(r)) + s'(r) \ln(r) + (b + b^2) s(r) \frac{b - 1 - \ln r}{r \ln r} \]
\[ = b \ln(r) s'(r) - b^2 s'(r) + s'(r) + (b + b^2) s'(r) \ln (s'(r)) + \ln(r) s'(r) + (b + b^2) s'(r) - (1 + b) s'(r) - (1 + b) \ln(r) s'(r) \]
\[ = (b + b^2) s'(r) \ln (s'(r)). \]

Hence, as claimed, \( \beta'_2(r) = s'(r) \log (s'(r)) = h(r) \log (h(r)) \).

\( \beta_2(1) \) is not defined. However with \( b > 1 \), one can easily see that \( \lim_{r \to 1^-} (\beta_2(r)) = C \). Hence \( \beta_2 = \beta_2(1 - \theta) \).

As \( H(r)^2 \geq R(r) \) for \( r \in [0, 1 - \theta] \), and \( H(r)^2 \leq R(r) \) for \( r \in [1 - \theta, 1] \), we have
\[ \gamma_H = \int_0^1 \min H(r)^2, R(r) dr = \gamma_1 + \gamma_2, \]
where
\[ \gamma_1 := \int_0^{1-\theta} R(r) dr = \int_0^{1-\theta} \left( \frac{r}{1-r} \right)^2, \]
\[ \gamma_2 := \int_{1-\theta}^1 H(r)^2 dr. \]
4. General k-SAT

It is easy to check that the antiderivative of $R(r)$ is

$$
\gamma_1(r) = 2 \ln(1 - r) + r + \frac{1}{1 - r} + C.
$$

Hence

$$
\gamma_1 = \gamma_1(1 - \theta) - \gamma_1(0) = 2 \ln \theta + \frac{1}{\theta} - \theta.
$$

To compute $\gamma_2$, let $\Gamma(a, z) := \int_z^\infty e^{-t} t^{a-1} dt$ be the (upper) incomplete gamma function. We claim that

$$
\int_0^r (-\ln(t))^b dt = \Gamma(1 + b, -\ln r) + C.
$$

We check

$$
\Gamma(1 + b, -\ln r) = \int_{-\ln r}^\infty e^{-t} t^b dt = \lim_{z \to \infty} \int_{-\ln r}^z e^{-t} t^b dt = \lim_{z' \to 0^+} \int_{-\ln z'}^{-\ln r} (-\ln (e^{-t}))^b (-e^{-t}) dt
$$

With $x = e^{-t}$, this is

$$
= \lim_{z' \to 0^+} \int_{z'}^r (-\ln(x))^b dx = \int_0^r (-\ln(x))^b dx.
$$

Hence the antiderivative of $H(r)^2$ is

$$
\gamma_2(r) = r - 2\Gamma(1 + b, -\ln r) \cdot a^b + \Gamma(1 + 2b, -\ln r) \cdot a^{2b} + C.
$$

and

$$
\gamma_2 = \gamma_2(1) - \gamma_2(1 - \theta).
$$

Note that the numerical calculation of $\gamma_2$ using the previous formula needs around 40 more significant digits than required of the output.

Numerical optimization yields that $\delta$ is maximized using $\theta \approx 0.5111885981$.

We obtain

$$
a \approx 1.1437170697,
b \approx 15.635592073,
\beta_H \leq 0.9062404894,
\gamma_H \geq 0.6122939734,
\delta_H \geq 0.0292762355,
\sigma(1 - \delta_H) \leq 0.4028867638.
$$

Hence using Theorem 4.21 gives the following result:
4.6. Optimized Distribution Functions for 3-SAT

Theorem 4.22 ([13])

\[ \Pr(\text{COMB}(F, s) \in \text{sat}(F)) \leq 2^{-0.4028867638n(F) - o(n(F))}. \]

Hence there is a randomized algorithm for 3-SAT with running time

\[ \mathcal{O}\left(1.32216^n(F)\right). \]
Chapter 5

Improved Analysis for 3-SAT and 4-SAT

In this chapter, we give the improved analysis using critical variables. We first show the general concepts we use. Then we give two new bounds for 3-SAT, one based on Rolf [13], and one based on the very recent improvement of Iwama et al. [4]. We also give a new bound for 4-SAT.

5.1 Critical Variables

Let $F$ be a satisfiable CNF. We call a variable $x \in V(F)$ critical if all satisfying assignments of $F$ agree on $x$. This means that if a variable is non-critical, then both $F[x \mapsto 0]$ and $F[x \mapsto 1]$ are satisfiable. By $V_C(F)$ we denote the set of critical variables of $F$. Let $n_c(F) := |V_C(F)|$, the number of critical variables of $F$. Let $c(F) := n_c(F)/n(F)$, the fraction of variables that are critical.

**Lemma 5.1** Critical variables are always nondefining, regardless of the assignment and subcube partition.

**Proof** Assume that a critical variable $x$ is defining for some $\alpha$ with subcube $B_\alpha$. Assume w.l.o.g. $\alpha(x) = 1$. Let $\beta$ be the assignment that agrees with $\alpha$ in all variables except for $x$ where $\beta(x) = 0$. As $x$ is defining for $B_\alpha$, $\beta$ must lie in a separate cube $B_\alpha'$ for a different satisfying assignment $\alpha'$. For $B_\alpha'$, $x$ must also be defining, as otherwise $\alpha \in B_\alpha'$. Hence $\alpha'(x) = 0$, a contradiction. □

The following simple approach reduces the number of variables of $F$ by one and preserves satisfiability with a probability depending on $c(F)$:

**Lemma 5.2** Let $F$ be a satisfiable CNF. Choose a variable $x$ u.a.r., and choose $\alpha(x)$ u.a.r. from $\{0, 1\}$ and let $G := F[x \mapsto \alpha(x)]$. Then $n(G) \leq n(F) - 1$, and with probability $p := 1 - c(F)/2$, $G$ is satisfiable.
5. Improved Analysis for 3-SAT and 4-SAT

Proof It is clear that \( n(G) \leq n(F) - 1 \). If \( x \notin V_C(F) \), then \( G \) is satisfiable by definition of a non-critical variable; this occurs with probability \( 1 - c(F) \). If \( x \in V_C(F) \), then \( G \) is satisfiable with probability \( 1/2 \); this occurs with probability \( c(F) \). Hence \( G \) is satisfiable with probability \( 1 - c(F) + c(F)/2 = 1 - c(F)/2 \).

For a \( k \)-SAT algorithm \( \mathcal{A} \), we denote by \( \text{Guess-} \mathcal{A} \) an algorithm that invokes \( \mathcal{A} \) \( n \) times, after \( \{0, 1, \ldots, n - 1\} \) guessing steps according to the previous lemma.

Definition 5.3 Let \( \mathcal{A} \) be a randomized algorithm that tries to find a satisfying assignment of a \((\leq k)\)-CNF. Algorithm \( \text{Guess-} \mathcal{A} \) repeats the following \( n \) times: Execute algorithm \( \mathcal{A} \). If this returns a satisfying assignment, return this together with the assignment produced so far. If this did not find a satisfying assignment, choose a variable u.a.r. and set it u.a.r. to 0 or 1.

At the end of \( \text{Guess-} \mathcal{A} \), we return the produced assignment.

The following theorem implies that if we want to achieve success probability of a \( k \)-SAT algorithm of \( \Omega(p^{n(F)} + o(n(F))) \), then we restrict our attention to formulas with \( c(F) \geq 2 - 2p \).

Theorem 5.4 Let \( 0 \leq c \leq 1 \) and \( 0 \leq p \leq 1 \). If \( \mathcal{A} \) is a randomized algorithm for \( k \)-SAT such that

\[
\Pr( \mathcal{A}(F) \in \text{sat}(F) ) \geq \Omega(p^{n(F)} + o(n(F)))
\]

for all satisfiable \((\leq k)\)-CNF formulas \( F \) with \( c(F) \geq c^* \), then

\[
\Pr( \text{Guess-} \mathcal{A}(F) \in \text{sat}(F) ) \\
\geq \min \left( \Omega \left( \left( 1 - \frac{c^*}{2} \right)^{n(F)} \right), \Omega \left( p^{n(F)} + o(n(F)) \right) \right), \tag{5.1}
\]

for all satisfiable \((\leq k)\)-CNF formulas \( F \). The running time of \( \text{Guess-} \mathcal{A} \) is at most the running time of \( \mathcal{A} \) times a polynomial in \( n \).

Proof We use induction on \( n(F) \). For \( n(F) = 0 \) or \( c(F) \geq c^* \), the statement is trivial. Otherwise, Lemma 5.2 tells us that after one guess step, we obtain \( F' \) with \( n(F') \leq n(F) - 1 \) s.t. \( F' \) is satisfiable with probability at least \( 1 - \frac{c}{2} \).

We have

\[
\Pr( \text{Guess-} \mathcal{A}(F) ) \geq \left( 1 - \frac{c^*}{2} \right) \Pr( \text{Guess-} \mathcal{A}(F') ).
\]

The statement now follows directly by inserting the induction hypothesis. \(\square\)
5.2 A Better Bound for PPSZ

We now use the Theorem 5.4 to prove a better bound for PPSZ. We want to prove the following theorem extending 4.6:

**Theorem 5.5** Let $F$ be a satisfiable $(\leq k)$-CNF. For any nice distribution function $H$, we have

$$\mathbb{P}_{\text{PPSZ}}(F, \alpha, s(n(F)) | B_a) \geq 2^{-\beta_H |D_a| + (1-\gamma_H)|N_a \setminus V_C(F)| + (1-R_k)|V_C(F)| - o(n(F))}.$$  

First note that the proof is similar to 4.6. We have to extend (4.4) to the following equation:

$$E[|\text{Forced}(F, \pi_{U,I}, \alpha, s|B_a)| \mid \pi_U \in \Gamma] \geq \gamma_H |N_a \setminus V_C(F)| + R_k |V_C(F)| - o(|N_a|).$$  

(5.2)

Remember that $p_F(x, F, \alpha, s|H) := \Pr(x \in \text{Forced}(F, \pi_{H,I}, \alpha, s))$, the probability that $x$ is forced given the assignment $\alpha$ and the random placement $\pi_{H,I}$. Remember that $\Gamma_{H,I}$ is the subset of placements we consider, and that $\pi_{H,I}$ is $\pi_{U,I}$ given $\pi_U \in \Gamma_{H,I}$. We have

$$E[|\text{Forced}(F, \pi_{U,I}, \alpha, s|B_a)| \mid \pi_U \in \Gamma_{H,I}] = E[|\text{Forced}(F, \pi_{H,I}, \alpha, s|B_a)|].$$

Now consider the critical clause tree we can build for some variable $x$. If $x$ is non-critical, then we might not be able to extend a node labeled with a defining variable, as the assignment $\alpha \oplus L$ might satisfy $F$. However, for a critical variable $x$, the critical clause tree can grow arbitrarily, as if $\{x\} \in L$, the assignment $\alpha \oplus L$ never satisfies $F$. Hence for critical variables $x$, we define $T_x$ to be a tree obtained from Lemma 3.15, as in unique $k$-SAT. For nondefining non-critical variables $x$, let $T_x$ be a tree obtained from Lemma 4.3, as before. Hence

$$E[|\text{Forced}(F, \pi_{H,I}, \alpha, s|B_a)|] \geq \sum_{x \in N} Q(T_x, \pi_{H,I}).$$

We now apply Lemma 4.13 as before and this is

$$\geq \sum_{x \in N_a} Q(T_x, \pi_{H}) - \lambda - o(1).$$

As $\lambda$ can be chosen arbitrarily small, we have

$$E[|\text{Forced}(F, \pi_{H,I}, \alpha, s|B_a)|] \geq \sum_{x \in N_a} Q(T_x, \pi_{H}) - o(1).$$  

(5.3)

We already know $Q(T_x, \pi_{H}) \geq \gamma_H - o(1)$, but for critical variables we get a better bound. It is easily seen that the requirement $H(r) \geq r$ implies $Q(T_x, \pi_{H}) \geq Q(T_x, \pi_{U,I})$. From the unique case we then know that $Q(T_x, \pi_{U,I}) \geq R_k - o(1)$. (5.2) follows now directly.
5.3 3-SAT

5.3.1 Our Improvement

We now combine Theorem 5.5 with Theorem 4.20 to obtain the following theorem very similar to Theorem 4.21:

**Theorem 5.6** Let $F$ be a satisfiable $(\leq k)$-CNF. Let $p \in [0.5, 1)$ be a constant, and let $c^* := 2 - 2p$.

Let $H$ be a nice distribution with $\beta_H \geq (1 - \gamma_H)(1 - c^*) - R\gamma H c^*$.

Then for

$$\delta_H = \frac{\sigma_k + \gamma_H(1 - c^*) + Rc^* - 1}{\sigma_k + \gamma_H + \beta_H - 1},$$

we have

$$\Pr(Guess-Comb(F, s) \in sat(F)) \geq 2^{\max\{-\log \rho \sigma_k(1 - \delta_H) n(F) - o(n(F))\}}.$$

We again only need this theorem for $k = 3$.

**Proof** The proof is very similar to the proof of Theorem 4.21. It is easily seen that if the required conditions hold, then $\delta_H$ is well-defined with $0 < \delta_H \leq 1$.

Using Theorem 5.4, we can assume $c(F) \geq c^*$.

We have by Theorem 5.5

$$p_{PPSZ}(F, \alpha, s(n(F)) | B_\alpha) \geq 2^{-\beta_H |D_k| + (1 - \gamma_H) |N_k\setminus V_C(F)| + (1 - R_k) |V_C(F)| - o(n(F))}$$

and by Theorem 4.20

$$\Pr(CHOENING(F, \beta) \in sat(F) \mid \beta \in B) \geq 2^{-\alpha_k |N| - o(n(F))}.$$

Solving the following equation for $\delta_H$ gives the fraction of defining variables for which PPSZ and CHOENING have the same probability:

$$\beta_H \delta_H + (1 - \gamma_H)(1 - \delta_H - c^*) + (1 - R)c^* = \sigma_k(1 - \delta_H).$$

We get

$$\beta_H \delta_H + 1 - \delta_H - c^* - \gamma_H + \gamma_H \delta_H + \gamma_H c^* + c^* - Rc^* = \sigma_k - \sigma_k \delta_H$$

and

$$\beta_H \delta_H - \delta_H + \gamma_H \delta_H + \sigma_k \delta_H = -1 + c^* + \gamma_H - \gamma_H c^* - c^* + Rc^* + \sigma_k$$

and hence

$$\delta_H = \frac{\sigma_k + \gamma_H(1 - c^*) + Rc^* - 1}{\sigma_k + \gamma_H + \beta_H - 1}.$$
For $|D_\alpha| \leq \delta_H n(F)$, as we required $\beta_H \geq 1 - \gamma_H$, the probability of PPSZ increases, and trivially for $|D_\alpha| \geq \delta_H n(F)$, the probability of SCHOENING increases.

Hence for $\alpha \in \text{sat}(F)$, we have
\[
\Pr \left( \text{COMB}(F,s) \in \text{sat}(F) \mid \beta \in B_\alpha \right) \geq 2^{-\sigma(1-\delta_H)n(F)-o(n(F))}.
\]
As this holds for all $\alpha \in \text{sat}(F)$, this implies the claim, by an argument similar to (4.1).

We use the $H$ of [13] as Section 4.6, but with different parameters. We now want to find optimal $\theta$ and $c$ to minimize the bound of the previous theorem. Numerical optimization yields that $\delta_H$ is maximized using $\theta \approx 0.52455825$ and $p \approx 0.756702705$. This gives
\[
c \approx 0.48659459,
a \approx 0.96782885577,
b \approx 7.19709520894,
\beta_H \leq 0.802563838,
\gamma_H \geq 0.607995502,
\delta_H \geq 0.03092732049,
\sigma(1-\delta_H) \leq 0.402201502.
\]
Hence using Theorem 5.6 gives the following result:

**Theorem 5.7**
\[
\Pr \left( \text{Guess-COMB}(F,s) \in \text{sat}(F) \right) \geq 2^{-0.402201502n(F)-o(n(F))}.
\]
There is a randomized algorithm for 3-SAT with running time
\[
O \left( 1.32153^n(F) \right).
\]

### 5.3.2 Combination with Improvement of [4]

In [4], Iwama et al. improved the algorithm from Iwama and Tamaki [5] by replacing SCHOENING with a new algorithm ISTTSch. We call the new combined algorithm ISTT. We derive from [4] that for $m^* \in [0,1]$, \[
\Pr(\text{ISTTSch}(F,\beta) \in \text{sat}(F) \mid \beta \in B_\alpha) \geq \min \left\{ 6^{-m^*n(F)}, \left( \frac{64}{63} \right)^{m^*n(F)} 1.28248358 \right\}.
\]
Note that this bound is not stated explicitly in [4]; Lemma 6 of [4] states a similar, but suboptimal bound. With this, we can now use the same approach as before to optimize for \( \theta, c^*, \text{ and additionally } m^* \). It is easily seen that we have to set \( m^* := -\log_6 \left( 1 - \frac{c^*}{2} \right) \). We find
\[
\theta \approx 0.5224565, \\
c^* \approx 0.4855858
\]
and obtain the following theorem:

**Theorem 5.8** There is a randomized algorithm for 3-SAT with running time \( \mathcal{O}(1.32065^{n(F)}) \).

### 5.4 4-SAT

We now consider 4-SAT. The best stated bound is from [5], \( \mathcal{O}(1.47390^{n(F)}) \). They used the old version of [8], however. Using the analysis for PPSZ from [13], we get \( \mathcal{O}(1.48981^{n(F)}) \). Using our improvement, we can improve this again.

We use \( H(r) := \min \{ 1, \frac{1}{2^r} \} \) for some \( \theta \in \left[ \frac{2}{3}, 1 \right] \). We have \( \beta_H = \frac{1}{5} \). Using Maple, we can give an explicit expression of \( R_4(r) \). For \( r \in \left[ 0, \frac{2}{3} \right] \), we have
\[
R_4(r) = \frac{1}{2} \frac{12r^2 - r - 1 + \sqrt{-3r^2 + 2r + 1}}{(r - 1)^2}. 
\]
For \( r \in \left[ \frac{2}{3}, 1 \right] \), we have \( R_4(r) = 1 \) by Lemma 3.33. We can now compute \( \gamma_H \) numerically.

We would like to use Theorem 5.6 now, but for \( \theta \) too large, the condition \( \beta_H \geq (1 - \gamma_H) \) gets violated. It turns out that the best we can achieve is setting \( \theta \) such that \( \beta_H = (1 - \gamma_H) \). In this case, the probability of PPSZ does not depend on \( d_\alpha \). We get \( \theta \approx 0.6803639 \), with \( \beta_H \leq 0.5556215010 \) and \( 1 - \gamma_H \leq 0.5556215122 \).

Using Theorem 5.5, we optimize for \( c^* \) and get \( c^* \approx 0.63878808 \). Hence we get a running time of \( \mathcal{O}(1.46928^{n(F)}) \).

This is already very close to the unique case \( \mathcal{O}(1.46899^{n(F)}) \). We do not know if it is possible to reach the unique case by using a more refined choice for \( H \), possibly depending on \( |D_\alpha| \), in combination with Schöning’s algorithm. The choice of \( H \) from [13] could be adapted for 4-SAT, but complex numerical computations and optimizations would be necessary.
Conclusion

We have seen how using a simple idea, the critical variables, the PPSZ algorithm can be improved for general 3-SAT and 4-SAT.

Schöning’s algorithm [14] with running time $O(1.3334^n)$ was the fastest known algorithm for 3-SAT at its publication, and it was extremely simple. The PPSZ algorithm [8] is somewhat faster for unique 3-SAT ($O(1.308^n)$). It is interesting how many improvement steps have been done to close that gap. First Schöning’s algorithm was improved alone, then combined with PPSZ. Finally some of the improvements for Schöning could be applied to the combination with PPSZ by Iwama et al. [4] to obtain $O(1.32113^n)$. We added another preprocessing step to achieve again a slightly better running time $O(1.32065^n)$. The best algorithm is now very complicated again, a combination of many different concepts. More improvements might be possible by combining these concepts more cleverly. Showing the unique case is the worst case in PPSZ and achieving running time $O(1.308^n)$ would make the fastest algorithm once more based on a simple idea. We have tried multiple approaches for that, but we have not obtained a useful result.

With the derandomization of Schöning’s algorithm by Moser and Scheder [6], the running time of deterministic algorithms is now rather close to running times of randomized algorithms. PPSZ has been derandomized in the unique case by Rolf [12], but no derandomization for general $k$-SAT is known. We think it is important to investigate general $k$-SAT better and attempt to improve or derandomize PPSZ, or give reasons why this cannot be done.

Note that the preprocessing step is very similar to PPSZ; it is easy to show that Guess-PPSZ has essentially the same success probability as PPSZ. However this does not hold for SCHÖNING. The problem is that the subcube dimension and with it the decision which of PPSZ and SCHÖNING should
be used is done after the necessary fraction of critical variables is ensured. We could not overcome this difficulty; hence we had to make the algorithm more complicated.

For 4-SAT, the gap between the general and the unique case is very small; the running times are $O(1.469278^n)$ and $O(1.468984^n)$. It is possible to adapt the distribution function $H$ to the dimension of the subcube. We could not use this with our choice of $H$ for 4-SAT, but with a better choice similar to the function from Rolf [13], it might be possible to close that gap.
Bibliography


