The Satisfiability Problem: Random Walk and Derandomization

Timo Eckstein

FAU Erlangen-Nürnberg

21. September 2014 - 3. October 2014

Outline

1 Introduction

2 Algorithm for solving k-SAT
- brute-force and splitting
- Random Walk algorithm by Uwe Schöning
  - The basic idea
  - Pseudocode
  - Estimate of the runtime
- Derandomization attempt by Dantsin et al.
  - The basic idea
  - Pseudocode
  - Estimate of the runtime
- Complete Derandomization by Moser & Scheder
  - The basic idea
  - Pseudocode
  - Estimate of the runtime

3 Conclusion

Introduction

- u literal $\Leftrightarrow$ $u{=}x$(variable) or $u = \bar{x}$ (negation of x)
- A finite set C of literals over pairwise distinct variables is called a clause.
- A finite set of clauses is called a formula in CNF (Conjunctive Normal Form). CNFs which have no more than k distinguishing literals are denoted ($\leq k$) CNF formulas, ones with exactly k literals k-CNF formulas.

Example

For a ($\leq$ 3)-CNF formula

$$(\bar{a} \lor \bar{b} \lor \bar{c}) \land (a \lor c) \qquad\qquad a, b, c \in \{0, 1\}$$

Introduction

- The task of deciding whether a CNF-formula F is satisfiable is labelled satisfiability problem (short: SAT)
- A mapping α: V $\rightarrow$ $\{$0,1$\}$ is called (truth) assignment.
- SAT is first NP-complete problem (shown by Stephen A.Cook)

brute-force and splitting

- assignment F is satisfiable iff $F|_{[x:=1]}$ or $F|_{[x:=0]}$ is
  $\Rightarrow$ recursive $O(2^n poly(n))$-algorithm
  $\Rightarrow$ upper bound for the runtime
- Especially for small k better runtimes possible: for instance for 3-SAT Rodošek achieved $O(1.476^n)$

Random Walk algorithm by Uwe Schöning

The basic idea

- applying a Monte Carlo approach onto k-SAT
- similarities:
  - same configuration space $\mathbb{Z}_2^n$
  - Choosing an initial element uniformly at random
- differences
  - Selection of the coordinate/atom/literal
  - "Flipping rule"

The basic idea

The probability that we do not find a satisfying assignment after t repetitions with independent random bits is

$$[1 - \mathbf{Pr}(N \leq 3n)]^t \leq e^{-\mathbf{Pr}(N \leq 3n)t}$$

Example

For an error probability less than 10^{-3}, you need $t := \frac{3 \cdot ln(10)}{\mathbf{Pr}(N \leq 3n)}$ independent repetitions of Schöning's algorithm.

Pseudocode

Schöning(($\leq$ k)-CNF formula F)

$\alpha \overset{\text{u.a.r.}}{\leftarrow} \{0,1\}^n$ // sample α **u**niformly **a**t **r**andom
return Schöning-Walk(F, α)

Schöning-Walk(($\leq$ k)-CNF formula F, assignment α)

for i = 0, .., t do
for i = 0, .., 3n do
if α satisfies F **then**
return α
else
C $\leftarrow$ any clause of F unsatisfied by α
u $\overset{\text{u.a.r.}}{\leftarrow}$ C // a random literal from C
$\alpha \leftarrow \alpha[u \mapsto 1]$
endif
endfor
return failure
endfor

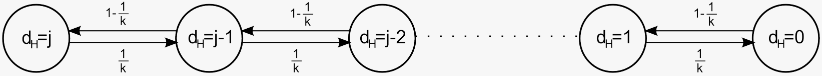
Estimate of the runtime

Theorem 1

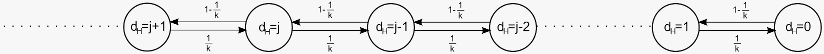
If the input k-SAT instance is satisfiably, then algorithm Schöning needs on expectation $O\left(\left(\frac{2(k-1)}{k}\right)^n\right)$ tries to find a satisfying assignment.

Proof

- WANTED : $\mathbf{Pr}(\exists t \leq 3n : Y_t = 0)$



- Consider the Markov chain:



- Let N_j the random variable that counts the number of steps until the first encounter of state 0

Estimate of the runtime

Lemma 2.2

For $q < \frac{1}{2}$ and $j \in \mathbb{N}_0$:

$$\mathbf{Pr}(N_j < \infty) = \left(\frac{q}{1-q}\right)^j$$

Proof

$$\begin{aligned}\mathbf{Pr}(N_j < \infty) &= \sum_{i=0}^{\infty} \binom{2i+j}{i} \cdot \frac{j}{2i+j} \cdot (1-q)^i \cdot q^{i+j} \\ &= q^j \cdot \sum_{i=0}^{\infty} \binom{2i+j}{i} \cdot \frac{j}{2i+j} \cdot (q \cdot (1-q))^i \\ &= q^j \cdot (B_2(q(1-q)))^j\end{aligned}$$

Estimate of the runtime

for $B_2(z)$ being the generalized Binomial series defined by

$$B_2(z) = \sum_i \binom{2i+1}{i} \cdot \frac{z^i}{2i+1} = \frac{1-\sqrt{1-4z}}{2z}$$

for which

$$(B_2(z))^r = \sum_i \binom{2i+r}{i} \cdot \frac{r}{2i+1} \cdot z^i$$

$\forall r \in \mathbb{N}_0$. So

$$\mathbf{Pr}(N_j < \infty) = q^j \cdot \left(\frac{1-\sqrt{1-4q+4q^2}}{2(1-q)q}\right)^j = q^j \cdot \left(\frac{1}{1-q}\right)^j$$

Estimate of the runtime

Lemma 2.3

For $q < \frac{1}{2}$ andl $j \in \mathbb{N}_0$ it holds:

$$\mathbf{E}(N_j \mid N_j > \infty) = \frac{j}{1-2q}$$

Proof

$$\mathbf{E}(N_j \mid N_j < \infty) = \frac{1}{\mathbf{Pr}(N_j < \infty)} \cdot \sum_{i=0}^{\infty} (2i+j) \cdot \binom{2i+j}{i} \cdot \frac{j}{2i+j} \cdot (1-q)^i \cdot q^{i+j}$$

$$\overset{\text{Lemma2.2}}{=} j \cdot (1-q)^j \cdot \sum_{i=0}^{\infty} (2i+j) \cdot \binom{2i+j}{i} \cdot (q \cdot (1-q))^i$$

$$= j \cdot (1-q)^j \cdot \frac{(B_2(q(1-q)))^j}{\sqrt{1-4q(1-q)}} = \frac{j}{1-2q}$$

Estimate of the runtime

Lemma 2.4

Let N the random variable that counts the number of steps until state 0 is encountered for the first time. For $q < \frac{1}{2}$ it holds, while n is still the number of variables:

$$\mathbf{Pr}(N < \infty) = \left(\frac{1}{2(1-q)}\right)^n$$

Proof

$$\mathbf{Pr}(N < \infty) = \sum_{j=0}^{n} \binom{n}{j} \cdot 2^{-n} \cdot \mathbf{Pr}(N_j < \infty)$$

$$\overset{\text{Lemma2.2}}{=} \sum_{j=0}^{n} \binom{n}{j} \cdot 2^{-n} \cdot \left(\frac{q}{1-q}\right)^j = \left(\frac{1}{2(1-q)}\right)^n$$

Estimate of the runtime

Lemma 2.5

For $q < \frac{1}{2}$ it holds:

$$\mathbf{E}(N \mid N < \infty) = \frac{qn}{1-2q}$$

Proof

$$\begin{aligned}
\mathbf{E}(N \mid N < \infty) &= \sum_i i \cdot \mathbf{Pr}(N = i \mid N < \infty) = \\
&= \sum_i i \cdot \sum_{j=0}^{n} \binom{n}{j} \cdot \mathbf{Pr}(N_j = i \mid N < \infty) = \\
&= \frac{2^{-n}}{\mathbf{Pr}(N < \infty)} \cdot \sum_{j=0}^{n} \binom{n}{j} \cdot \sum_i i \cdot \mathbf{Pr}(N_j = i) =
\end{aligned}$$

Estimate of the runtime

$$= \frac{2^{-n}}{\mathbf{Pr}(N < \infty)} \cdot \sum_{j=0}^{n} \binom{n}{j} \cdot \mathbf{E}(N_j \mid N_j < \infty) \cdot \mathbf{Pr}(N_j < \infty)$$

$$\stackrel{\text{Lemma2.2,2.3,2.4}}{=} (1-q)^n \cdot \sum_{j=0}^{n} \binom{n}{j} \cdot \frac{j}{1-2q} \cdot \left(\frac{q}{1-q}\right)^j$$

$$= \frac{n \cdot (1-q)^n}{1-2q} \cdot \sum_{j=0}^{n} \binom{n-1}{j-1} \cdot \left(\frac{q}{1-q}\right)^j$$

$$= \frac{n \cdot (1-q)^n}{1-2q} \cdot \frac{q}{1-q} \cdot \left(1 + \frac{q}{1-q}\right)^{n-1} = \frac{nq}{1-2q}$$

Estimate of the runtime

Lemma 2.6

For $q < \frac{1}{2}$ and $\lambda \geq 1$ it holds:

$$\mathbf{Pr}\left(N \leq \frac{\lambda q n}{1-2q}\right) > \left(1 - \frac{1}{\lambda}\right) \cdot \left(\frac{1}{2(1-q)}\right)^{n}$$

Proof:
Write μ for $\mathbf{E}(N \mid N < \infty)$.Observe that

$$\mathbf{Pr}(N > \lambda\mu \mid N < \infty) < \frac{1}{\lambda}$$

by Markov's inequality, and
$\mathbf{Pr}(N \leq \lambda\mu) = \mathbf{Pr}(N > \lambda\mu \mid N < \infty) \cdot \mathbf{Pr}(N < \infty)$

Estimate of the runtime

Now using $q = \frac{1}{k}, k \geq 3$ and $\lambda = 3$, we obtain

$$\mathbf{Pr}(\exists t \leq 3n : Y_t = 0) = \mathbf{Pr}(N \leq 3n) > \frac{2}{3}\left(\frac{k}{2(k-1)}\right)^n$$

whereas, for a satisfiable formula, the expected number of repetitions of procedure Schöning until a satisfying assignment is found is at most $\frac{1}{\mathbf{Pr}(N \leq 3n)}$:

$$\frac{3}{2}\left(\frac{2(k-1)}{k}\right)^n$$

Derandomization attempt by Dantsin et al.

The basic idea

- $d_H(\alpha, \beta) := |\{x \in V | \alpha(x) \neq \beta(x)\}|$ with α, β truth assignments is called Hamming distance
- $B_r(\alpha) := \{d_H(\alpha, \beta) \leq r\}$ is denoted Hamming ball, with volume $vol(n, r) := |B_\alpha(r)| = \sum_{i=0}^{r} \binom{n}{i}$.
- $C \subseteq \{0, 1\}^n$ covering code of radius r and length n $\Leftrightarrow \cup_{\alpha \in C} B_r(\alpha) = \{0, 1\}^n$
- Ball-k-SAT: Decide whether $B_r(\alpha)$ contains a satisfying assignment

Pseudocode

cover-search(($\leq$ k)-CNF formula F over n variables)

$r := \frac{n}{k+1}$
construct a covering code C of radius r and $|C| \leq \frac{2^n}{\binom{n}{k}} poly(n)$
return $\bigcup_{\alpha \in C} sat - searchball(F, \alpha, r)$

$sat - searchball((\leq k)$-CNF formula F, assignment α, radius r)

if α satisfies F **then**
    return true
elseif r = 0 **then**
    return false
else
    C $\leftarrow$ any clause of F unsatisfied by α
    return $\bigcup_{u \in C} sat - searchball(F|_{u:=1}, \alpha, r - 1)$
endif

Correctness of this algorithm

Initial step

- $r = 0 \Rightarrow B_0(\alpha) = \{\alpha\}$

Induction step $(r - 1 \rightarrow r)$

- Let be α^* a satisfying assignment with $d_H(\alpha, \alpha^*) \leq r$ and C the selected clause
- Let $\alpha' := \alpha^*[u := 0]$
- We observe that $d_H(\alpha, \alpha') \leq r - 1$ and α' satisfies $F|_{u:=1}$(induction hypothesis).

Estimate of the runtime

Lemma 2.12

The algorithm sat-searchball solves BALL-k-SAT in time $O(k^r poly(n))$.

Proof:

If F is a $(\leq k)$-CNF formula, then each call to searchball causes at most k recursive calls.

Estimate of the runtime

Theorem 2

Suppose some algorithm A solves BALL-k-SAT in O(a^rpoly(n)) steps. Then there is an algorithm B solving k-SAT in time $O\left(\left(\frac{2a}{(a+1)}\right)^n poly(n)\right)$, and B is deterministic if A is.

Proof

Lemma 2.13

For all n $\in \mathbb{N}$, $0 \leq r \leq n$, every code C of covering radius r and length n has at least $\frac{2^n}{vol(n,r)}$ elements. Furthermore, there is such a C with

$$|C| \leq \frac{2^n poly(n)}{vol(n,r)},$$

and furthermore, C can be constructed deterministically in time O($|C|$ poly(n)).

Proof: This Lemma is the case k=2 of upcoming Lemma 2.19.

Estimate of the runtime

Lemma 2.14

For $0 \leq \rho \leq \frac{1}{2}$ and t $\in \mathbb{N}$, it holds that

$$\binom{t}{\rho t} \geq \frac{1}{\sqrt{8t\rho(1-\rho)}} \left(\frac{1}{\rho}\right)^{\rho t} \left(\frac{1}{1-\rho}\right)^{(1-\rho)t}$$

Set r $:= \frac{n}{(a+1)}$ and construct a covering code C of radius r and length n and call A(F, α, r) for each $\alpha \in$ C.

$$|C| a^r poly(n) \overset{\text{Lemma2.13}}{\leq} \frac{2^n a^r poly(n)}{vol(n,r)} \overset{\text{Lemma2.14}}{\leq} \frac{2^n a^{\frac{n}{a+1}} poly(n)}{(a+1)^{\frac{n}{a+1}} \left(\frac{a+1}{a}\right)^{\frac{na}{a+1}}} =$$

$$= \left(\frac{2a}{a+1}\right)^n poly(n)$$

Complete Derandomization by Moser & Scheder

The basic idea

- further development of Dantsin et al.'s covering code idea
- Promise-Ball-k-SAT:
  Ball-k-SAT+ promise that $B_r(\alpha)$ contains a satisfying assignment
- k truth values instead of 2
  - $vol^{(k)}(t,r) := |B^{(k)}(w)| = \binom{t}{r}(k-1)^r$, w $\in \{1,...,k\}^t$
  - Let t $\in \mathbb{N}$. A set C$\subseteq \{1,..,k\}^t$ is called k-ary covering code of radius r $\Leftrightarrow \bigcup_{w\in C} B_r^{(k)}(w) = \{1,..,k\}^t$

Pseudocode

searchball − fast($k \in \mathbb{N}$, ($\leq$ k)-CNF formula F, assignment α, radius r,
code C$\subseteq \{1, ..., k\}^t$)

if α satisfies F **then**
    return true
elseif r $= 0$ **then**
    return false
else
    G $\leftarrow$ a maximal set of pairwise disjoint k-clauses of F unsatisfied by α
    if $|G| < t$ **then**
        return $\bigcup_{\beta \in \{0,1\}^{vbl(G)}} sat - searchball(F|_{[\beta]}, \alpha, r)$
    else
        H $\leftarrow \{C_1, ..., C_t\} \subseteq$ G
        return $\bigcup_{w \in C} searchball - fast(k, F, \alpha[H, w], r - (t - 2t/k), C)$
    endif
endif

Estimate of the runtime

Lemma 2.19

$\forall t, k \in \mathbb{N}$ and $0 \leq s \leq \frac{t}{2}$. $\exists$ C $\subseteq \{1, ..., k\}^t$ of covering radius s such that:

$$|C| \leq \left\lceil \frac{\ln(k) \cdot k^t \cdot t}{\binom{t}{s} \cdot (k-1)^s} \right\rceil =: m$$

and furthermore, C can be constructed deterministically in time O($|C|$ poly(t)).

Proof:

$$\mathbf{Pr}\left[w' \notin \bigcup_{w \in C} B_s^{(k)}(w)\right] = \left(1 - \frac{vol^{(k)}(t,s)}{k^t}\right)^{|C|} < e^{-|C| \frac{vol^{(k)}(t,s)}{k^t}}$$

$$\leq e^{-t \cdot \ln(k)} = k^{-t}$$

Estimate of the runtime

runtime of constructing such a covering code

- t:= $\lfloor \ln(n) \rfloor$

$$O(|C| \cdot poly(t)) \leq O(k^t \cdot poly(t)) = O(n^{\ln(k)} \cdot ploy(\ln(n))) = O(poly(n))$$

Estimate of the runtime

Lemma 2.20

With the aid of Lemma 2.14 and Lemma 2.19 we get the following approximation of $|C|$ which we will need later. Let ρ be $\frac{1}{k}$:

$$\binom{t}{\frac{t}{k}} \geq \frac{1}{\sqrt{8}} \cdot k^{\frac{t}{k}} \cdot \left(\frac{k}{k-1}\right)^{\frac{(k-1)\cdot t}{k}} = \frac{k^t}{\sqrt{8t} \cdot (k-1)^{\frac{(k-1)t}{k}}}$$

So we obtain, for t sufficiently large:

$$|C| \leq \left\lceil \frac{\ln(k) \cdot k^t \cdot t}{\binom{t}{s} \cdot (k-1)^s} \right\rceil \leq \frac{t^2 \cdot k^t \cdot (k-1)^{\frac{(k-1)t}{k}}}{k^t \cdot (k-1)^{\frac{t}{k}}} = t^2 \cdot (k-1)^{t - \frac{2t}{k}}$$

Estimate of the runtime

Case m $<$ t

```
6:  G ← a maximal set of pairwise disjoint k-clauses of F unsatisfied by α
7:  if |G| < t then
8:      return ∨_{β∈{0,1}^{vbl(G)}} sat-searchball(F|_[β], α, r)
```

Lemma 2.21

If every clause in F that is not satisfied by α has size at most k - 1, then sat-searchball(F, α, r) runs in time O($(k-1)^r$poly(n)).

Proof:

Since every unsatisfied clause and every $F|_{u:=1}$ hade at most k - 1 literals, the algorithm calls itself at worst k – 1 times.

Estimate of the runtime

Case m < t

$$2^{km}O((k-1)^r poly(n)) \leq O(2^{kt}(k-1)^r poly(n)) \leq$$
$$\leq O(2^{ln(n)k}(k-1)^r poly(n)) \leq O((k-1)^r poly(n))$$

$\overset{\text{Theorem 2}}{\Rightarrow}$ runtime k-SAT: $O\left(\left(\frac{2(k-1)}{k}\right)^n\right)$

Estimate of the runtime

Case m $\geq$ t

```
10:          H ← {C_1, ..., C_t} ⊆ G
11:          return ∨_{w∈C} searchball-fast(k,F,α[H,w], r-(t-2t/k),C)
```

- Let α[H,w] be the assignment obtained from α by flipping the value of the w_i^{th} literal
- Define w* $\in \{1, \ldots, k\}^t$ as follows: for each 1$\leq$ i $\leq$t set w_i^* to j such that α^* satisfies the jth literal in C_i.

Observe:

- There is some $w \in \{1, ..., k\}^t$ such that $d_H(\alpha[w^*], \alpha^*) = d_H(\alpha, \alpha^*) - t$
- Let w, w′ $\in \{1, ..., k\}^t$. Then $d_H(\alpha[w], \alpha[w\prime]) = 2d_H(w, w\prime)$

Estimate of the runtime

Case m $\geq$ t

Lemma 2.24

If α^* is a satisfying assignment of F, then there is some w $\in$ C such that $d_H(\alpha[w], \alpha^*) \leq d_H(\alpha, \alpha^*)$ - t + 2s. In particular, if $B_r(\alpha)$ contains a satisfying assignment, then there is some w $\in$ C such that $B_{r-t+2s}(\alpha[w])$ contains it, too.

Proof:

C has covering radius s $\Rightarrow d_H(w, w^*) \leq s$

$\overset{\text{Observation 2}}{\Rightarrow} \; d_H(\alpha[w], \alpha[w^*]) \leq 2s$

$$d_H(\alpha[w], \alpha^*) \leq d_H(\alpha[w^*], \alpha^*) + d_H(\alpha[w], \alpha[w^*]) \leq$$
$$\leq d_H(\alpha, \alpha^*) - t + 2s$$

Estimate of the runtime

Case m $\geq$ t

- $|C|$ recursive calls
- decrease of the complexity parameter r by $t - 2s = t - \frac{2t}{k}$ in each step

$$|C|^{\frac{r}{t-\frac{2t}{k}}} \overset{\text{Lemma 2.20}}{\leq} \left(t^2 \cdot (k-1)^{t-\frac{2t}{k}}\right)^{\frac{r}{t-\frac{2t}{k}}} = \left(t^{\frac{2}{t-\frac{2t}{k}}} \cdot (k-1)\right)^r$$

$$= (k-1)^{r+o(n)}$$

$\overset{\text{Theorem 2}}{\Rightarrow}$ runtime k-SAT: $\mathrm{O}\left(\left(\frac{2(k-1)}{k}\right)^{n+o(n)}\right)$

Theorem 3

There is a deterministic algorithm solving k-SAT in time $O\left(\frac{2(k-1)}{k}\right)^{n+o(n)}$

U. Schöning:
A probabilistic algorithm for k-SAT based on limited local search and restart; Algorithmica 32 (2002) 615-623

R.A. Moser, D. Scheder:
A Full Derandomization of Schöninger's k-SAT Algorithm; in Proc. 43rd ACM Symp. On Theory of Computing (STOC), pp. 245-251, 2011

D. Scheder:
Algorithms and Extremal Properties of SAT and CSP, dissertation, Swiss Federal Institute of Technology Zurich, 2011

E. Dantsin, A. Goerdt, E.A. Hirsch, R. Kannan, J. Kleinberg, C. Papadimitriou, P.Raghavan, U. Schöning:
A deterministic $(2 - \frac{2}{k+1})^n$ algorithm for k-SAT based on local search. Theoretical Computer Science, Volumne 289, Issue 1, 23 October 2002, Page 69-83

F. J. MacWilliams and N. J. A. Sloane. The theory of error-correcting codes. II. North- Holland Publishing Co., Amsterdam, 1977. North-Holland Mathematical Library, Vol. 16.