Finding Satisfying Assignments by Random Walk

Rolf Wanka, Erlangen

Overview

- Preliminaries
- A Randomized Polynomial-time Algorithm for 2-SAT
- A Randomized $O(2^n)$-time Algorithm for 3-SAT
- A Randomized $O((4/3)^n)$-time Algorithm for 3-SAT

Preliminaries (I)

Satifiability problem SAT: Given a Boolean formula Φ in Conjunctive Normal Form (CNF) over n variables $x_1, \ldots, x_n$ and m clauses.

CNF = Conjunction of clauses;
Clause = Disjunction of literals;
Literal = variable or negation of variable

Question: Is there a truth assignment to the variables such that Φ evalutes to TRUE ?

Example for $n = 4$ and $m = 5$:

$$\Phi = (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (x_1 \vee x_2) \wedge (x_4 \vee \bar{x}_3) \wedge (x_4 \vee \bar{x}_1)$$

Satisfied by

$$x_1 := \text{TRUE};\ x_2 := \text{TRUE};\ x_3 := \text{FALSE};\ x_4 := \text{TRUE}$$

$k \in \mathbb{N}$: For k-SAT, Φ is restricted to that each clause has exaclty k literals.

So,

$$\Phi = (x_1 \vee \bar{x}_2) \wedge (\bar{x}_1 \vee \bar{x}_3) \wedge (x_1 \vee x_2) \wedge (x_4 \vee \bar{x}_3) \wedge (x_4 \vee \bar{x}_1)$$

is an instance of 2-SAT.

Time complexity:

SAT is NP-complete.

3-SAT is NP-complete

2-SAT is in P.

A Randomized Polynomial-time Algorithm for 2-SAT (I)

2-SAT Algorithm ($c \in \mathbb{N}$ being an arbitrary constant):

- Start with an arbitrary truth assignment;
- Repeat up to $2cn^2$ times, terminating if all clauses are satified the following iteration:
  - Choose an arbitrary clause C that is not satisfied;
  - Choose uniformly at random one of the literals in C and switch the value of its variable;
- If a valid truth assignment has been found, return YES
- Otherwise, return NO.

Theorem: Φ is satifiable $\Rightarrow$ Pr(algo. returns YES)$\geq 1 - \frac{1}{2^c}$

A Randomized Polynomial-time Algorithm for 2-SAT (II)

Let S represent a satisfying assignment.
A_i: the truth assignment after the ith iteration.
X_i: number of variables in A_i with identical value in S

Algorithm terminates with YES if $X_i = n$.

We have

$$\begin{aligned} \Pr(X_{i+1} = 1 \mid X_i = 0) &= 1 \\ \Pr(X_{i+1} = j+1 \mid X_i = j) &\geq \frac{1}{2} \\ \Pr(X_{i+1} = j-1 \mid X_i = j) &\leq \frac{1}{2} \end{aligned}$$

A Randomized Polynomial-time Algorithm for 2-SAT (III)

Let S represent a satisfying assignment.
A_i: the truth assignment after the ith iteration.
X_i: number of variables in A_i with identical value in S

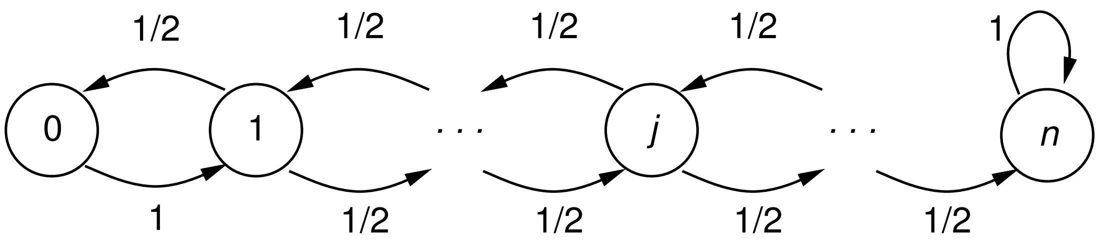
Algorithm terminates with YES if $X_i = n$.

We have

$$\begin{aligned}
\Pr(X_{i+1} = 1 \mid X_i = 0) &= 1 \\
\Pr(X_{i+1} = j+1 \mid X_i = j) &= \frac{1}{2} \\
\Pr(X_{i+1} = j-1 \mid X_i = j) &= \frac{1}{2}
\end{aligned}$$

A Randomized Polynomial-time Algorithm for 2-SAT (IV)

Graphical representation



h_j = expected no. of steps to reach n when starting from j

We have the system of equations:

$$\begin{aligned} h_n &= 0 \\ h_j &= \tfrac{1}{2} \cdot (h_{j-1} + h_{j+1}) + 1 \qquad \text{for } j \in \{1, \ldots, n-1\} \\ h_0 &= h_1 + 1 \end{aligned}$$

Its unique solution: $h_j = n^2 - j^2$

A Randomized Polynomial-time Algorithm for 2-SAT (V)

That means (if Φ is satisfiable, S the only valid assignment):

The expected number of iterations until the algorithm returns YES is at most n^2.

The algorithm executes $2cn^2$ iterations.
Divide the iterations into c segments $\Sigma_1, \ldots, \Sigma_c$ of $2n^2$ iterations each.
Let Z_i be the number of iterations from the start of Σ_i until S is found.
Then by Markov's inequality,

$$\Pr(Z_i \geq 2n^2) \leq \frac{E[Z_i]}{2n^2} \leq \frac{n^2}{2n^2} = \frac{1}{2}$$

$\Rightarrow \Pr(\text{algo. fails to find } S) \leq \left(\frac{1}{2}\right)^c$

A Randomized $O(2^n)$-time Algorithm for 3-SAT (I)

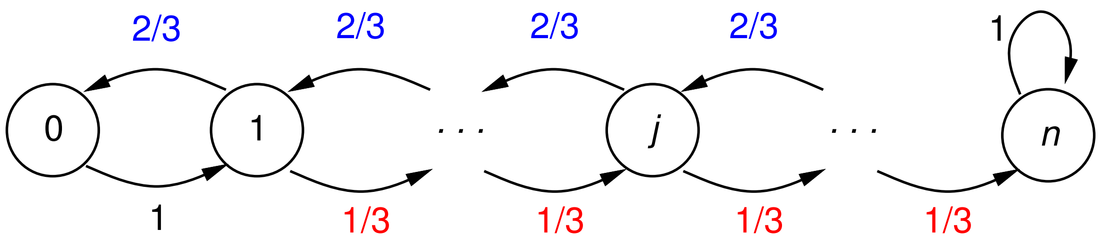
First 3-SAT Algorithm:

– Start with an arbitrary truth assignment;
– Repeat up to ℓ times, terminating if all clauses are satified the following iteration:
  - Choose an arbitrary clause C that is not satisfied;
  - Choose uniformly at random one of the literals in C and switch the value of its variable;
– If a valid truth assignment has been found, return YES
– Otherwise, return NO.

Theorem: Φ is satifiable $\Rightarrow$ The expected no. ℓ of iterations to find a valid truth assignment is $\Theta(2^n)$.

A Randomized $O(2^n)$-time Algorithm for 3-SAT (II)

Graphical representation assuming satisfying assignment S and counting the "correct" variables



h_j = expected no. of steps to reach n when starting from j

We have the system of equations:

$$\begin{aligned} h_n &= 0 \\ h_j &= \frac{2}{3} \cdot h_{j-1} + \frac{1}{3} \cdot h_{j+1} + 1 \qquad \text{for } j \in \{1, \ldots, n-1\} \\ h_0 &= h_1 + 1 \end{aligned}$$

Its unique solution: $h_j = 2^{n+2} - 2^{j+2} - 3(n-j)$

A Randomized $O(2^n)$-time Algorithm for 3-SAT (III)

Observations:

1. If A_0 is chosen u. a. r, X_0 follows a symmetric binomial distribution,

$$\Pr(X_0 = j) = \binom{n}{j} \cdot \left(\frac{1}{2}\right)^n$$

with $E[X_0] = \frac{1}{2}n$. I. e., there is an exponentially small but non-negligible probability that A_0 matches S in significantly more than $\frac{1}{2}n$ variables.

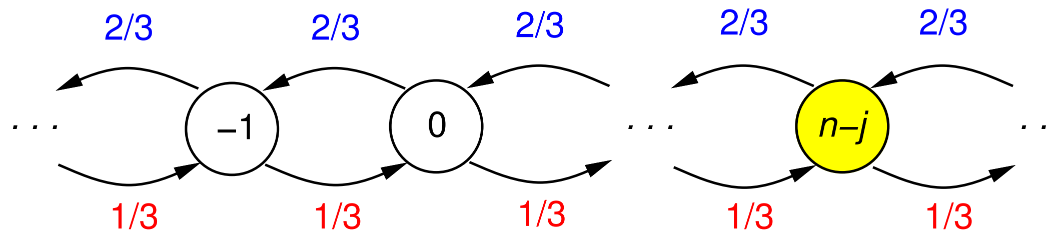
2. The algorithm is more likely to move towards 0 than towards n. The longer we run, the more likely we have moved towards 0.

A Randomized $O((4/3)^n)$-time Algorithm for 3-SAT (I)

Schöning's 3-SAT Algorithm:

- Repeat up to ℓ times, terminating if all clauses are satified:
  - (a) Start with a truth assignment chosen u. a. r.; [Restart]
  - (b) Repeat the following up to $3n$ times terminating if all clauses are satified:
    - (1) Choose an arbitrary clause C that is not satisfied;
    - (2) Choose uniformly at random one of the literals in C and switch the value of its variable;
- If a valid truth assignment has been found, return YES
- Otherwise, return NO.

A Randomized $O((4/3)^n)$-time Algorithm for 3-SAT (II)

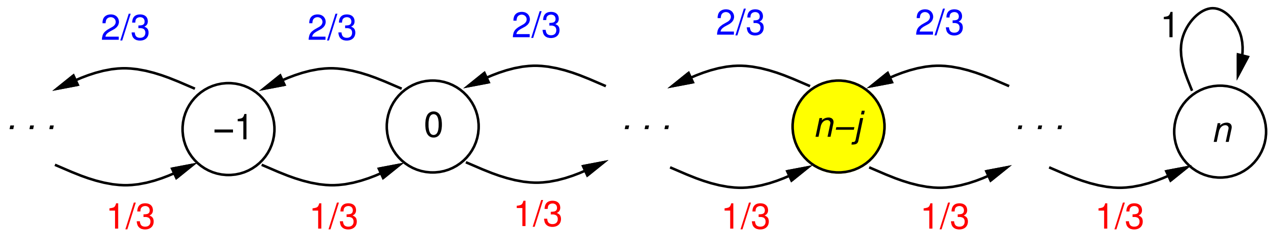


The probability of exactly k moves down and $k+j$ moves up in a sequence of $j+2k$ moves:

$$\binom{j+2k}{k} \cdot \left(\frac{2}{3}\right)^k \left(\frac{1}{3}\right)^{j+k}$$

A Randomized $O((4/3)^n)$-time Algorithm for 3-SAT (III)

q_j = (lower bound on) the probability that Schöning's algorithm reaches n when it starts with an assignment with exactly j mismatches.



So,

$$q_j \geq \max_{k \in \{0,\ldots,j\}} \binom{j+2k}{k} \cdot \left(\frac{2}{3}\right)^k \left(\frac{1}{3}\right)^{j+k}$$

In particular,

$$q_j \geq \binom{3j}{j} \cdot \left(\frac{2}{3}\right)^j \left(\frac{1}{3}\right)^{2j}$$

A Randomized $O((4/3)^n)$-time Algorithm for 3-SAT (IV)

By Stirling's Formula:

$$\binom{3j}{j} = \frac{(3j)!}{j! \cdot (2j)!} \geq \frac{\sqrt{2\pi(3j)}}{4\sqrt{2\pi j} \cdot \sqrt{2\pi(2j)}} \cdot \left(\frac{3j}{e}\right)^{3j} \cdot \left(\frac{e}{2j}\right)^{2j} \cdot \left(\frac{e}{j}\right)^{j}$$

$$= \underbrace{\frac{\sqrt{3}}{8\sqrt{\pi}}}_{=:a} \cdot \frac{1}{\sqrt{j}} \cdot \left(\frac{27}{4}\right)^{j}$$

So,

$$q_j \geq a \cdot \frac{1}{\sqrt{j}} \cdot \frac{1}{2^j}$$

and $q_0 = 1$.

A Randomized $O((4/3)^n)$-time Algorithm for 3-SAT (V)

Let q denote the probability that Schöning's algorithm reaches n in $3n$ steps.

$$\begin{aligned} q &\geq \sum_{j=0}^{n} \Pr(X_0 = n - j) \cdot q_j \\ &\geq \frac{1}{2^n} + \sum_{j=1}^{n} \binom{n}{j} \left(\frac{1}{2}\right)^n \cdot a \cdot \frac{1}{\sqrt{j}} \cdot \frac{1}{2^j} \\ &\geq \frac{a}{\sqrt{n}} \cdot \left(\frac{3}{4}\right)^n \end{aligned}$$

Hence, the expected overall number of assignments tried is $1/q = O(\sqrt{n} \cdot (4/3)^n) = o(1.33333334^n)$.

Further Results

Iwama/Tamaki & Rolf: $O(1.32216^n)$

Schmitt/W.: $O(1.322030^n)$

Algorithm is a hybrid (running also the other known algorithms) that also swaps from time to time all values of the variables.