Randomized SAT algorithms

Martin Babka

March 16, 2011

History

Randomized algorithms

- Paturi, Pudlák, Saks and Zanez (PPSZ) algorithm solves unique 3-SAT in $\mathcal{O}(1.3071^n)$ time.
- Schöning proposed $\mathcal{O}(\text{poly}(n)(4/3)^n)$ algorithm for any satisfiable 3-SAT formula.
- Iwama and Tamaki improved it to $\mathcal{O}(1.3238^n)$. Refined analysis of PPSZ improved the bound to $\mathcal{O}(1.32266^n)$.
- The best known result, $\mathcal{O}(1.32216^n)$, is by Rolf from 2006.

Deterministic algorithms

- PPSZ has already been derandomized.
- In 2010 Moser and Scheder showed a full derandomization of Schöning's k-SAT Algorithm.

Probability basics

Markov inequality

Let X be a non-negative random variable and $k > 0$. Then

$$\mathbf{Pr}\left(X \geq k\mathbf{E}\left[X\right]\right) \leq \frac{1}{k}.$$

Geometric distribution

$X \approx \mathrm{Ge}(p)$ if $\mathbf{Pr}\left(X = k\right) = (1-p)^{k-1}p$. Hence $\mathbf{E}\left[X\right] = \frac{1}{p}$.

Random walk

- We are given a digraph with the set of nodes being equal to all possible assignments of variables.
- Edges are determined by the algorithms.
- We calculate the probability of reaching a satisfiable assignment from a random one.

2-SAT, a simple example

Algorithm

Let $c \in \mathbb{N}$ be an arbitrary constant and n be the number of variables of the given formula.

Algorithm

- Repeat up to c times.
  - Start with an arbitrary assignment.
  - Repeat up to $2n^2$ times:
    - 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**.
- Return **NO**.

If the formula is satisfiable, then $\mathbf{Pr}\left(\mathbf{YES}\right) \geq 1 - 2^{-c}$.

2-SAT, a simple example

Analysis of random walk

Fix a satisfiable solution S.

- State j represents the assignments having Hamming distance j from S, they differ in j variables when compared to S.
- Random walk around states $0, \ldots, n$.
- The value h_j denotes the expected number of steps to reach 0 when in j.

For our random walk we have that

- $h_0 = 0$,
- $h_n = 1 + h_{n-1}$,
- $h_j = 1 + \frac{1}{2} h_{j+1} + \frac{1}{2} h_{j-1}$ hence $h_{j+1} = 2h_j - h_{j-1} - 2$.

Solution of the system of linear equations is $h_j = 2nj - j^2 \leq n^2$.

2-SAT, a simple example

Analysis

What is the probability of finding a solution in $\mathcal{O}\left(n^2\right)$ steps?

- We start in a state j, it is chosen at random.
- The expected number of steps to find S is at most n^2 .
- We repeat the iteration $2n^2$ steps.
- By Markov inequality $\mathbf{Pr}\left(\text{not finding } S\right) \leq \frac{1}{2}$.
- Because of c independent restarts the overall probability of not finding a satisfying solution is at most 2^{-c}.

We have a randomized polynomial algorithm for 2-SAT with a negligible error. The situation changes dramatically for k-SAT, $k > 2$, why?

3-SAT

The same algorithm

What is the expected number of steps to reach the state 0?

- $h_0 = 0$.
- $h_j = 1 + \frac{1}{3}h_{j-1} + \frac{2}{3}h_{j+1}$ hence $h_{j+1} = \frac{3}{2}h_j - \frac{1}{2}h_{j-1} - \frac{3}{2}$.
- $h_n = 1 + h_{n-1}$.

The unique solution is $h_j = 2^{n+2} - 2^{n-j+2} - 3j$.

- We are likely to run towards the state n than to the state 0.
- The expected number of steps is exponential and so is the expected running time of the algorithm.
- The complexity for the error probability 2^{-c} is $\mathcal{O}(c\,\mathrm{poly}(n)2^n)$.
- We want a lower base.

k-SAT

Idea

Notation

- We assume that we have a formula with n variables.
- Let t be a parameter – the number of restarts.

Idea

- It is likely to run towards the state n during a random walk.
- Make the random walks shorter.
- Repeat random walks, do restarts, (exponentially) many times.
- The probability that the algorithm never finishes in the state 0 is exponentially low with respect to the number of restarts, t.

Algorithm

- Repeat up to t times.
  - Start with an arbitrary assignment.
  - If a valid truth assignment has been found, return **YES**.
  - Repeat up to $3n$ times:
    - 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**.
- Return **NO**.

- We need to find a suitable t.
- The c loop from 2-SAT may be simulated by ct restarts.

k-SAT

Analysis

- Fix a satisfying solution S.
- States are the same as in case of 2-SAT; j denotes the number of variables having different values in S.
- Let q_j be the probability of reaching the state 0 when starting in the state j.

Estimating q_j

- Moreover we allow i steps backwards (towards n). Now we need $j + i$ step towards 0.
- Exact analysis using Catalan numbers. Simpler analysis permits „negative“ states.

Estimating q_j

- $q_j \geq \max_{i \in \{0,\ldots,j\}} \binom{j+2i}{i} \left(\frac{1}{k}\right)^{j+i} \left(\frac{k-1}{k}\right)^{i}$.
- The value $\binom{j+2i}{i}$ equals the number of paths going $j+i$ steps towards 0 and i steps towards n.
- The above estimate is valid because we use maximum.
- Because $j \leq i$ we do not consider more than $3n$ steps.
- Choose $i \approx \frac{j}{k-2}$ and then $q_j \geq \Omega\left(j^{-2} \cdot \left(\frac{1}{k-1}\right)^{j}\right)$.
- For $k=3$ using Stirling approximation it may be shown that $q_j = \Omega\left(\frac{1}{\sqrt{j}} \cdot 2^{-j}\right)$.

- Let p be the probability of reaching 0 in one restart.

$$p = \sum_{j=0}^{n} \mathbf{Pr}\left(\text{starting in } j\right) \cdot q_j .$$

- $\mathbf{Pr}\left(\text{starting in } j\right) = \binom{n}{j}\left(\frac{1}{2}\right)^n$.
- Thus $p = 2^{-n} \cdot \Omega\left(n^{-2}\right) \cdot \left(1 + \frac{1}{k-1}\right)^n$.
- In one restart we find a solution with probability at least p.
- From the expected value of geometric distribution we need at least $t = \frac{2}{p}$ restarts to find it with the probability at least 0.5.
- Another c repetitions lower the error rate to 2^{-c}.

k-SAT

Result for k-SAT

- We need $\mathcal{O}\left(n^2\left(1-\frac{1}{k}\right)^n\right)$ restarts for a constant error.
- For large values of k, $k=\Omega(n)$, we are not far from 2^n.

Result for 3-SAT

- We need $\mathcal{O}\left(\sqrt{n}\left(\frac{4}{3}\right)^n\right)$ restarts to have a constant error.
- The overall complexity of the algorithm is $\mathcal{O}\left(\mathrm{poly}(n)\left(\frac{4}{3}\right)^n\right)$.

Other applications

- The same approach also works in CSP.
- The best algorithm is a simple combination of PPSZ and Schöning's algorithms. Analysis is far more complicated.

Literature

- Schöning, U.: A Probabilistic Algorithm for k-SAT and Constraint Satisfaction Problems, 1999
- Rolf, D.: Improved Bound for the PPSZ/Schöning-Algorithm for 3-SAT, 2006
- Moser, A., R., Scheder, D.: A Full Derandomization of Schöning's k-SAT Algorithm, 2010
- Rolf Wanka's presentation
- Luca Trevisan lecture notes on Randomized Algorithms