Advanced Algorithmics

Strategies for Tackling Hard Problems

Sebastian Wild

Markus Nebel

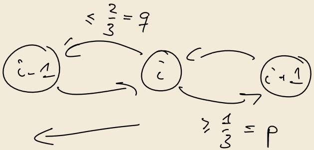
Lecture 16

2017-06-12

$$\Pr[\#\text{iterations} \geq 2k^2] \leq \frac{1}{2}$$

certainty independent repetitions $\rightsquigarrow \left(\frac{1}{2}\right)^{\text{certainty}}$ $\square$.

Try the same idea for 3SAT.



$$a = \frac{q}{p} \qquad b = -\frac{c}{p}, \quad c = 1$$

$$\dot{y}_i = -1 \cdot 2^i - 3 \sum_{j=1}^{i} 2^{i-j}$$
$$= -2^i - 3(2^i - 1)$$
$$= -4 \cdot 2^i + 3$$

$$y_0 = -\sum_{\ell=0}^{k-1} \dot{y}_\ell = 4(2^k - 1) - 3k$$

$\geq 2 y_0 \quad = 8 \cdot 2^k$ iterations

worse than brute-force method!

$$\dot{y}_i = \left(\prod_{j=1}^{i} a_j\right) \cdot \underset{-1}{\underset{\shortparallel}{\dot{y}_0}} + \sum_{j=1}^{i} \left(\prod_{k=j+1}^{i} a_k\right) b_j$$

Schöning's Randomized 3SAT Algorithm

procedure Schöning3SAT(ϕ,$certainty$):
    k = number of variables **in** ϕ
    for $i = 1, \ldots, certainty \cdot 24 \left\lceil \sqrt{k} \left(\frac{4}{3}\right)^k \right\rceil$ **do**
        Choose assignment $\alpha \in \{0,1\}^k$ uniformly at random.
        for $j = 1, \ldots, 3k$ **do**
            if α fulfills ϕ **return** "ϕ satisfiable"
            Arbitrarily choose a clause $C = \ell_1 \vee \ell_2 \vee \ell_3$ that is not satisfied under α.
            Choose ℓ from $\{\ell_1, \ell_2, \ell_3\}$ uniformly at random.
            α = assignment obtained by negating ℓ.
    return "ϕ probably not satisfiable"

Theorem 4.53 (Schöning3SAT is OSE-MC for 2SAT)

Let ϕ be a 3SAT formula with n clauses over k variables.

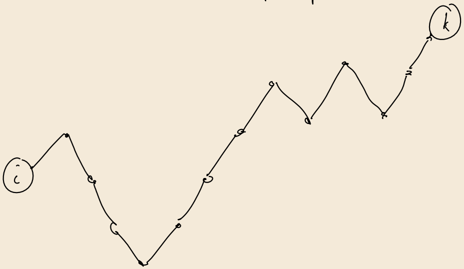
1. If ϕ is unsatisfiable, `Schöning3SAT` always returns "probably not satisfiable".
2. If ϕ is satisfiable, `Schöning3SAT` returns "satisfiable" with probability $\geq 1 - 2^{-certainty}$.
3. `Schöning3SAT` runs in time $\mathcal{O}\left(certainty \cdot k^{3/2} \left(\frac{4}{3}\right)^k n\right)$.

Proof: ϕ satisfiable $\Rightarrow$ α^* satisfying assignment

Lower bound for success probability of one local-search round

$\to$ random starting state i $\qquad i \overset{\mathcal{D}}{=} \mathrm{Bin}(k, \frac{1}{2})$

Need $k - i =: u$ up-steps to k $\qquad \Pr[i] = \binom{k}{i} \left(\frac{1}{2}\right)^k$



length of such path $= u + 2\ell$

$q_i :=$ prob to reach k in $\leq 3k$ steps

$u + 2\ell \leq 3k \qquad \ell \leq k \qquad \boxed{\ell = u}$

$q_i \geq$ prob to reach k in $3u$ steps

$\geq$ prob $2u$ of the $3u$ steps go up

*Extend all runs that reach k after less than 3u steps to sequences of up and down steps of length exactly 3u; prob to reach k in at most 3u steps is at least as large as prob to be *above k* after *exactly* 3u steps, which happens iff we have at least 2u up-steps among the 3u steps.*

$$= \binom{3u}{u} \left(\frac{2}{3}\right)^{u} \left(\frac{1}{3}\right)^{2u}$$

$$\binom{3u}{u} = \frac{(3u)!}{u!\,(2u)!}$$

$$\geq \frac{\sqrt{2\pi\, 3u}}{2\sqrt{2\pi u} \cdot 2\sqrt{2\pi\, 2u}} \cdot \frac{\left(\frac{3u}{e}\right)^{3u}}{\left(\frac{u}{e}\right)^{u} \left(\frac{2u}{e}\right)^{2u}}$$

$$= \underbrace{\frac{\sqrt{3}}{8\sqrt{2\pi}}}_{c} \cdot \frac{1}{\sqrt{u}} \cdot \left(\frac{3^3}{2^2}\right)^{u}$$

Stirling

$$n! \in (1,2) \cdot \sqrt{2\pi n} \left(\frac{n}{e}\right)^{n}$$

$$q_i \geq \frac{c}{\sqrt{u}} \cdot \left(\frac{\cancel{3}^3 \cdot \cancel{2}}{2^{\cancel{2}} \cdot \cancel{3} \cdot \cancel{3}^2}\right)^{u} = \frac{c}{\sqrt{u}} \cdot 2^{-u}$$

$$q_k = 1$$

$$q_i := \Pr[\text{one round is successful}] = \sum_{i=0}^{k} \Pr[(i)] \cdot q_i$$

$$u = k-i$$

$$\geq \frac{1}{2^k} + \sum_{u=1}^{k} \binom{k}{u} \left(\frac{1}{2}\right)^{k} \cdot \frac{c}{\sqrt{u}}\, 2^{-u}$$

$$\geq \frac{c}{\sqrt{k}} \left(\frac{1}{2}\right)^k \cdot \underbrace{\sum_{j=0}^{k} \binom{k}{j} \left(\frac{1}{2}\right)^j \cdot 1^{k-j}}_{\left(\frac{1}{2}+1\right)^k} \qquad \text{Binomial Theorem}$$

$$= \frac{c}{\sqrt{k}} \left(\frac{3}{4}\right)^k$$

$$\mathbb{E}[\#\text{ trials}] = 1/q \leq \frac{8\sqrt{2\pi}}{\sqrt{3}} \left(\frac{4}{3}\right)^k \sqrt{k}$$

$2 \cdot$ $\rightsquigarrow$ prob $\geq \frac{1}{2}$

$\square$

Smart probability amplification: Karger's Min-Cut

Definition 4.54 (Min-Cut)

Given: A (multi)graph $G = (V, E, c)$, where $c : E \to \mathbb{N}$ is the multiplicity of an edge

Feasible Solutions: cuts of G, i.e., $M(G) = \{(V_1, V_2) \ : \ V_1 \cup V_2 = V \ \wedge \ V_1 \cap V_2 = \emptyset\}$,

$V_1 \neq \emptyset \neq V_2$

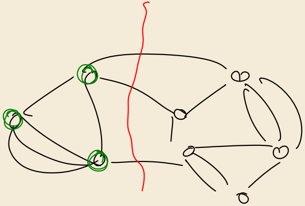
Goal: Minimize

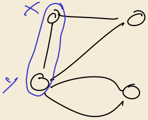
Costs: $\sum_{e \in C(V_1,V_2)} c(e)$, where $C(V_1, V_2) = \left\{\{u, v\} \in E \ : \ u \in V_1 \wedge v \in V_2\right\}$.

best deterministic algos in

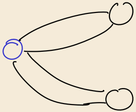
$O\left(n \cdot m \cdot \left(\log\left(\frac{n^2}{m}\right) + 1\right)\right)$

w.c. $\rightsquigarrow$ $O(n^3)$





G



$G/\{x,y\}$

Random Contraction

procedure contractionMinCut($G = (V, E, c)$) // assume G connected

  Set $label(v) := \{v\}$ **for** every vertex $v \in V$.

  while G has more than 2 vertices

    Choose random edge $e = \{x, y\} \in E$. proportional to c

    $G := Contract(G, e)$. G/e

    Set $label(z) := label(x) \cup label(y)$ **for** z the vertex resulting from x and y.

  Let $G = (\{u, v\}, E', c')$; **return** $(label(u), label(v))$ with cost $c'(\{u, v\})$.

Theorem 4.55 (contractionMinCut correct with some probability)

`contractionMinCut` is a poly-time randomized algorithm that finds a minimal cut for a given multigraph G with n vertices with probability $\geq 2/(n(n-1))$.

Obvious: returns feasible solution

n-2 contractions, can be implemented in $O(n^2)$

Optimal Solution? C cut

Observation: C returned iff no edge of C was contracted

Fix one optimal cut C

E_i = ith contraction does not use edge in C

$E_1, \dots, E_{n-2}$

$\bigcap_{i=1}^{n-2} E_i$ = C returned $\rightsquigarrow$ optimal

E_i's not independent

$$\Pr\left[\bigcap_{i=1}^{n-2} E_i\right] = \Pr[E_1] \cdot \Pr[E_2 \mid E_1] \cdot \dots \Pr\left[E_{n-2} \,\middle|\, \bigcap_{i=1}^{n-3} E_i\right]$$

$k = \text{cost}(C)$ = # edges (including parallel ones)

$\Rightarrow$ $m \geq \frac{nk}{2}$ every node must have deg $\geq k$

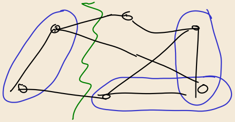
m includes parallel edges

$$\Pr[E_1] = \frac{m-k}{m} = 1 - \frac{k}{m} \geq 1 - \frac{k}{\frac{k \cdot n}{2}} = 1 - \frac{2}{n}$$

G/F_i after first $i-1$ contractions $\rightsquigarrow$ $n-i+1$ vertices

$$\Pr\left[E_i \mid \bigcap_{i=1}^{i-1} E_i\right] \longrightarrow F_i \cap C = \emptyset$$

C is still a min-cut in G/F_i



- contraction removes possible cuts, but does not allow new ones
- value of cuts in G/F_i = value of that cut in G

$\rightsquigarrow$ $\text{cost}(C) = k$

G/F_i has at least $\frac{k\cdot(n-i+1)}{2}$ edges

m_i = #edges in G/F_i

n_i = #vertices "

$$\Pr\left[E_i \mid \bigcap_{i=1}^{i-1} E_i\right] = \frac{m_i - k}{m_i} \geq 1 - \frac{k}{\frac{k(n-i+1)}{2}}$$

$$= 1 - \frac{2}{n-i+1}$$

$$\Pr\left[\bigcap_{i=1}^{n-2} E_i\right] \geq \prod_{i=1}^{n-2}\left(1-\frac{2}{n-i+1}\right)$$

$$= \prod_{\ell=3}^{n} \frac{\ell-2}{\ell} = \frac{1\cdot 2}{n(n-1)} = \frac{1}{\binom{n}{2}} \qquad \square$$

$\binom{n}{2}$ repetitions, return best cut

$\Omega(n^2)$ repetitions, $\Theta(n^2)$ implementation.

↳ Try not to run till end.