Advanced Algorithmics

Strategies for Tackling Hard Problems

Sebastian Wild

Markus Nebel

Lecture 17

2017-06-15

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

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

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

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

↳ $\binom{n}{2}$ indep. rep. $\qquad \left(1 - \frac{1}{\binom{n}{2}}\right)^{\binom{n}{2}} < \frac{1}{e}$

$$\left(1 - \frac{1}{n}\right)^n \to e^{-1}$$

↳ Try not to run full cut.

Lemma 4.56 (Threshold for contractionMinCut)

Let $l : \mathbb{N} \to \mathbb{N}$ a monotonic, increasing function with $1 \leq l(n) \leq n$. If we stop `contractionMinCut` whenever G only has $l(n)$ vertices and determine for the resulting graph G/F deterministically a minimal cut, then we need time in

$O(n^3)$

$$\mathcal{O}(n^2 + l(n)^3)$$

and we find a minimal cut for G with probability at least

$$\frac{\binom{l(n)}{2}}{\binom{n}{2}}$$

Proof: Prob of success $\Pr\{\text{return min cut}\} = \Pr\{\bigcap_{i=1}^{n-l(n)} E_i\}$

$$\geq \prod_{i=1}^{n-l(n)} \left(1 - \frac{2}{n-i+1}\right)$$

$$= \frac{\prod_{i=1}^{n-2} \left(1 - \frac{2}{n-i+1}\right)}{\prod_{i=n-l(n)+1}^{n-2} \left(1 - \frac{2}{n-i+1}\right)}$$

Running time ✓

(one contraction + determ. run on $l(n)$)

$$= \frac{1 / \binom{n}{2}}{1 / \binom{\ell(n)}{2}} = \frac{\binom{\ell(n)}{2}}{\binom{n}{2}} \qquad \square$$

Since $\frac{n^2}{\ell(n)^2} \geq \frac{\binom{n}{2}}{\binom{\ell(n)}{2}}$ $\rightsquigarrow$ $\frac{n^2}{\ell(n)^2}$ indep. repetitions succeed with prob. $\boxed{> 1 - \frac{1}{e}}$

Running $O\left((n^2 + \ell(n)^3) \cdot \frac{n^2}{\ell(n)^2} \right) = O\left(\underbrace{\frac{n^4}{\ell(n)^2}}_{\alpha} + \underbrace{n^2 \ell(n)}_{\beta} \right) = O(n^{8/3})$

$= o(n^3)$

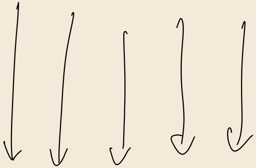
Want $\alpha = \Theta(\beta)$

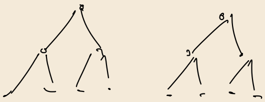
$\ell(n) = \Theta(n^{2/3})$ $\qquad \ell(n) = \lfloor n^{2/3} \rfloor$

$4 - \frac{4}{3} = \frac{8}{3}$

Can we do better?

- initial choices of edges mostly good ← don't repeat
- later choices are more likely to fail ← try many times





Karger's Min-Cut Improved

procedure KargerSteinMinCut($G(V, E, c)$)
    $n = |V|$
    if $n \geq 6$
        compute minimal cut deterministically simple
    else
        $h = \left\lceil 1 + \frac{n}{\sqrt{2}} \right\rceil^{\geq 2}$ $\rightsquigarrow$ reduce size by factor $\frac{1}{\sqrt{2}}$
        G/F_1 = Contract random edges **in** G until h nodes left
        $(C_1, cost_1)$ = KargerSteinMinCut(G/F_1)
        G/F_2 = Contract random edges **in** G until h nodes left
        $(C_2, cost_2)$ = KargerSteinMinCut(G/F_2)
        if $cost_1 < cost_2$ **return** $(C_1, cost_1)$ **else** $C_2, cost_2)$

Theorem 4.57 (KargerSteinMinCut beats deterministic min-cut)

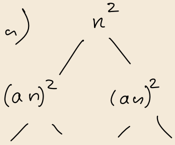
`KargerSteinMinCut` runs in time $\mathcal{O}(n^2 \cdot \log(n))$ and finds a minimal cut with probability $\Omega(\frac{1}{\log(n)})$.

Proof: Running time upper bound $T(n)$

$$T(n) = O(1) \qquad n \leq 6$$

$$T(n) = 2 \cdot T(a \cdot n) + O(n^2) \qquad a \approx \frac{1}{\sqrt{2}}$$

$$T(n) = O(n^2 \log n)$$



n^2

$2 \cdot a^2 n^2 = n^2$

$\log_{\sqrt{2}}(n)$

Success Prob.: P_ℓ = probability that G/F_i $(i = 1, 2)$ of size $\left\lceil 1 + \frac{\ell}{\sqrt{2}} \right\rceil$ still contain min cut C

$$\geq \frac{\binom{\lceil 1 + \ell/\sqrt{2} \rceil}{2}}{\binom{\ell}{2}} = \frac{(1 + \ell/\sqrt{2})(1 + \ell/\sqrt{2} - 1)}{\ell(\ell - 1)} \geq \frac{1}{2}$$

$Succ(n)$ = prob. KargerStein MinCut(n) returns min cut C

Then $P_e \cdot Succ(\lceil 1 + \frac{e}{\sqrt{2}} \rceil)$ is prob. that one of the two runs returns C

$\left(1 - P_e \cdot Succ(\lceil 1 + \frac{e}{\sqrt{2}} \rceil)\right)^2$ prob that both runs fail

$Succ(2) = 1$

$$Succ(e) \geq 1 - \left(1 - P_e \cdot Succ\left(\frac{1}{\sqrt{2}} e\right)\right)^2$$

$$\geq 1 - \left(1 - \frac{1}{2} \cdot Succ\left(\frac{1}{\sqrt{2}} e\right)\right)^2$$

$\rightsquigarrow$ One can show that every solution of recurrence is in $\Theta\left(\frac{1}{\log n}\right)$ $\square$.

$\Rightarrow$ $O(\log n)$ repetitions $\rightsquigarrow$ constant success prob.

$O(\log^2 n)$ repetitions $\rightsquigarrow$ prob $\to 1$ $(n \to \infty)$

running time $O(n^2 \log^3 n)$ still better $O(n^3)$

5 Approximation Algorithms and Inapproximability

Optimization Problems: NPO

- each valid instance x has non-empty set of feasible solutions $M(x)$
- objective function cost assigns to $y \in M(x)$ an <u>integral</u> value
- check in poly-time
  - x valid instance
  - $y \in M(x)$
  - $\mathrm{cost}(y)$

optimization versions asks for an optimal solution

evaluation problems — " — optimal objective value

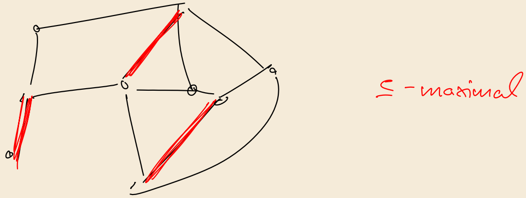
5.1 Introductory Examples

Vertex-Cover NP-complete

Definitions (Matching)

$G = (V, E)$

- $M \subseteq E$ matching $\Leftrightarrow$ no two edges in M share a vertex
- matching saturated or $\subseteq$-maximal $\Leftrightarrow$ no addition possible
- matching maximal if $\nexists$ larger matching



Saturated matching easy to compute

Add an $e \in E$ to M

for each $e \in E$

try to add e

until no more additions

Maximal Matchings

tougher to compute

blossoms-algorithm

Any matching M

proves that minimal vertex cover C needs $|M|$ nodes.

Proof: $e \in M$ $\quad e = \{u, v\}$

C has to contain u or v (has to cover e)

disjoint $\rightsquigarrow$ one node per $e \in M$

Definition 5.1 (Approximation Ratio)
Let $U = (\Sigma_I, \Sigma_O, L, L_I, M, cost, goal)$ an optimization problem and algorithms A be consistent with U. For every $x \in L_I$ we define the *relative error* $\varepsilon_A(x)$ *of A on x* by

$$\varepsilon_A(x) = \frac{|cost(A(x)) - Opt_U(x)|}{Opt_U(x)}$$

and the *approximation ratio* $R_A(x)$ *of A on x* by

$$R_A(x) \quad = \quad \max\left\{\frac{cost(A(x))}{Opt_U(x)}, \frac{Opt_U(x)}{cost_A(x)}\right\} \quad = \quad 1 + \varepsilon_A(x).$$

2-approx $\rightsquigarrow$ $\varepsilon \leq 1$ $\rightsquigarrow$ always at most twice as bad as the optimal solution

Theorem 5.2 (Matching is 2-approx for Vertex Cover)

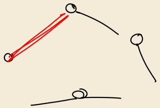
Before algorithm is a 2-approximation algorithm for problem Vertex-Cover.

Algo A: Compute a $\subseteq$-maximal matching M in G

$\rightsquigarrow$ Pick both vertices of the edges in M

Proof: Output is VC:

Let $e \in E$ $\rightsquigarrow$ To show $u \in C$, $v \in C$
$e = \{u,v\}$



Since M is saturated, every edge $e \in E$ has one endpoint touched by M (otherwise could add e to M)

$\Rightarrow |M| \leq OPT_{VC}$

$cost_A = 2|M|$

$$\Rightarrow r = \frac{cost}{opt} = \frac{2|M|}{OPT} \leq \frac{2|M|}{|M|} = 2$$

$\square$

Immediate Questions:

- better analysis for approximation ratio

  No $\rightsquigarrow$ tight examples (below)

- improve algorithm (e.g. maximal matching)

  No $\rightsquigarrow$

- Different idea for a lower bound for VC?

  Open Problem

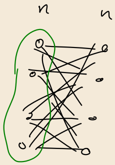
  poly-time

  - no constant-approx better 2 known
  - proven unless $P = NP$ no 1.3... -approx possible
  - conjecture implies 2-approx optimal

    open

Tight Example:

- Complete bipartite graph



  ≤-maximal matching has n edges

  $\Rightarrow$ A $2n$ nodes in VC (all nodes ...)

  VC with n nodes exists.

  $\Rightarrow$ in some graphs, one node per edge suffices

- Same lower bound: K_n complete graph for n odd

  $\Rightarrow$ maximal matching has size $\frac{n-1}{2}$

  but size of optimal VC is $n-1$

$\Rightarrow$ in other graph, no single matched node may be left out.

Remarks:

- Vertex-Cover NP-complete
- Maximal-Matching $\in P$

- VC in NP Yes-certificate : VC
- MM in NP Yes-certificate : M

- No-certificates

(a) bipartite graphs |maximal matching| = |minimal VC|
(later LP duality)

$\Rightarrow$ No-certificates for both problems

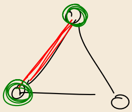
- VC = maximal matching Yes-certificate

$\exists$ VC with size $\leq k$ No-certificate is any matching of size $> k$

$\Rightarrow$ on bipartite graphs VC, MM $\in NP \cap \text{co-}NP$

$\in P$

(b) G not bipartite



min VC > max matching in general