Maximal independent set

trivial sequential algorithm

• inherently sequential

• from node point of view: each thinks can join MIS if others stay out

• randomization breaks this symmetry

Randomized idea

• each node joins with some probability

• all neighbors excluded

• many nodes join

• few phases needed

Algorithm:

• all degree 0 nodes join

• node $v$ joins with probability $1/2d(v)$

• if edge $(u,v)$ has both ends marked, unmark lower degree vertex

• put all marked nodes in IS

• delete all neighbors

Intuition: $d$-regular graph

• vertex vanishes if it or neighbor gets chosen

• mark with probability $1/2d$

• prob (no neighbor marked) is $(1 − 1/2d)^d$, constant

• so const prob. of neighbor of $v$ marked—destroys $v$

• what about unmarking of $v$’s neighbor?

• prob(unmarking forced) only constant as argued above.

• So just changes constants

• const fraction of nodes vanish: $O(\log n)$ phases

• Implementing a phase trivial in $O(\log n)$.

Idea of staying marked applies to general case: prob. chosen for IS, given marked, exceeds $1/2$
• suppose \( w \) marked. only unmarked if higher degree neighbor marked
• higher degree neighbor marked with prob. \( \leq 1/2d(w) \)
• only \( d(w) \) neighbors
• prob. any superior neighbor marked at most \( 1/2 \).

How about prob. neighbor gets marked?
• Define \textbf{good} vertices: at least \( 1/3 \) neighbors have lower degree

  • Intuition: good means “high degree”
  • Prob. lower degree neighbor marked exceeds \( 1/2d(v) \)
  • prob. no neighbor of good marked \( \leq (1 − 1/2d(v))d(v)/3 \leq e^{-1/6} \).
  • So some neighbor marked with prob. \( 1 − e^{-1/6} \)
  • Stays marked with prob. \( 1/2 \)
  • deduce prob. good vertex killed exceeds \( (1 − e^{-1/6})/2 \)
  • Problem: perhaps only one good vertex?

Good edges
• Idea: since “high degree” vertices killed, means most edges killed
• any edge with a good neighbor
• has const prob. to vanish
• show half edges good
• deduce \( O(\log n) \) iterations.

Proof
• Let \( V_B \) be bad vertices; we count edges with both ends in \( V_B \).
• direct edges from lower to higher degree \( d_i \) is indegree, \( d_o \) outdegree
• if \( v \) bad, then \( d_i(v) \leq d(v)/3 \)
• deduce

\[
\sum_{V_B} d_i(v) \leq \frac{1}{3} \sum_{V_B} d(v) = \frac{1}{3} \sum_{V_B} (d_i(v) + d_o(v))
\]

• so \( \sum_{V_B} d_i(v) \leq \frac{1}{2} \sum_{V_B} d_o(v) \)
which means indegree can only “catch” half of outdegree; other half must go to good vertices.

more carefully,

- \( d_o(v) - d_i(v) \geq \frac{1}{3}(d(v)) = \frac{1}{3}(d_o(v) + d_i(v)) \).
- Let \( V_G, V_B \) be good, bad vertices
- degree of bad vertices is

\[
2e(V_B, V_B) + e(V_B, V_G) + e(V_G, V_B) = \sum_{v \in V_B} d_o(v) + d_i(v)
\]

\[
\leq 3 \sum (d_o(v) - d_i(v))
\]

\[
= 3(e(V_B, V_G) - e(V_G, V_B))
\]

\[
\leq 3(e(V_B, V_G) + e(V_G, V_B))
\]

Deduce \( e(V_B, V_B) \leq e(V_B, V_G) + e(V_G, V_B) \). result follows.

Derandomization:

- Analysis focuses on edges,
- so unsurprisingly, pairwise independence sufficient
- prob vertex marked \( 1/2d \)
- neighbors \( 1, \ldots, d \) in increasing degree order
- Let \( E_i \) be event that \( i \) is marked.
- Let \( E'_i \) be \( E_i \) but no \( E_j \) for \( j < i \) (makes disjoint events so can add probabilities)
- \( A_i \) event no neighbor of \( i \) chosen
- Then prob eliminate \( v \) at least

\[
\sum \Pr[E'_i \cap A_i] = \sum \Pr[E'_i] \Pr[A_i | E'_i] \\
\geq \sum \Pr[E'_i] \Pr[A_i]
\]

\( (E'_i \) just forces some neighbors not marked so increases bound)\)
- But expected marked neighbors \( 1/2 \), so by Markov \( \Pr[A_i] > 1/2 \)
- so prob eliminate \( v \) exceeds \( \sum \Pr[E'_i] = \Pr[\cup E_i] \)
- lower bound as \( \sum \Pr[E_i] - \sum \Pr[E_i \cap E_j] = 1/2 - d(d - 1)/8d^2 > 1/4 \)
- so \( 1/2d \) prob. \( v \) marked but no neighbor marked, so \( v \) chosen
- Wait: show \( \Pr[A_i | E'_i] \geq \Pr[A_i] \)
- true if independent
- not obvious for pairwise, but again consider $d$-uniform case
- measure
  \[
  \Pr[\neg A_i \mid E'_i] \leq \sum \Pr[E_w \mid E'_i] \quad \text{(sum over neighbors $w$ of $i$)}
  \]
- measure
  \[
  \begin{align*}
  \Pr[E_w \mid E'_i] &= \frac{\Pr[E_w \cap E'_i]}{\Pr[E'_i]} \\
  &= \frac{\Pr[(E_w \cap \neg E_1 \cap \cdots) \cap E_i]}{\Pr[(\neg E_1 \cap \cdots) \cap E_i]} \\
  &= \frac{\Pr[\neg E_1 \cap \cdots \mid E_i]}{\Pr[E_w \mid E_i]} \\
  &\leq \frac{\Pr[E_w]}{1 - \sum_{j \leq i} \Pr[E_j \mid E_i]} \\
  &= 2 \Pr[E_w]
  \end{align*}
  \]
  (last step assumes $d$-regular so only $d$ neighbors with odds $1/2d$)

- Generate pairwise independent with $O(\log n)$ bits
- try all polynomial seeds in parallel
- one works
- gives deterministic $NC$ algorithm

with care, $O(m)$ processors and $O(\log n)$ time (randomized)
LFMIS P-complete.

Project

Dates

- Classes end 12/13, wednesday
- Final homework due 12/12, tuesday
- Project due 12/8 (MIT restriction)

Options

- Reading project
  - Read some hard papers
  - Write about them more clearly than original
  - graded on delta
– best source: STOC/FOCS/SODA

• Implementation project
  – read some randomized algorithms papers,
  – implement
  – develop interesting test sets
  – identify hard cases
  – devise heuristics to improve

• In your work:
  – use a randomized algorithm in your research;
  – write about it

**MST**

**Review Background**

• kruskal
• boruvka
• verification

**Intuition:** “fences” like selection algorithm.

**Sampling theorem:**

• Heavy edges
• pick $F$ with probability $p$
• get $n/p$ $F$-heavy edges

**Recursive algorithm without boruvka:**

$$T(m, n) = T(m/2, n) + O(m) + T(2n, n) = O(m + n \log n)$$

(sloppy on expectation on $T(2n,n)$)

**Recursive algorithm with 3 boruvka steps:**

$$T(m, n) = T(m/2, n/8) + c_1(m + n) + T(n/4, n/8)$$

$$\leq c(m/2 + n/8) + c_1(m + n) + c(n/4 + n/8)$$

$$= (c/2 + c_1)m + (c/8 + c_1 + c/4 + c/8)n$$

$$= (c/2 + c_1)(m + n)$$

so set $c = 2c_1$ (not sloppy expectation thanks to linearity).

**Notes:**

• Chazelle $m \log \alpha(m, n)$ via relaxed heap
• Ramachandran and Peti optimal deterministic algorithm (runtime unknown)
• open questions.
**Minimum Cut**

deterministic algorithms

- Max-flow
- Gabow

Min-cut implementation

- data structure for contractions
- alternative view—permutations.
- deterministic leaf algo

Recursion:

\[
\begin{align*}
p_{k+1} & = p_k - \frac{1}{4} p_k^2 \\
qu_k & = \frac{4}{p_k} + 1 \\
qu_{k+1} & = q_k + 1 + \frac{1}{q_k}
\end{align*}
\]