Date midterm.

Treaps

Review:

- Dictionaries for **ordered** sets
- Binary tree.
- Tree balancing by rotations
- drawbacks in geometry: rebuild on rotation

Returning to average case:

- Assign random "arrival orders" to keys
- Build tree **as if** arrived in that order
- Average case applies
- No rotations on searches

Choosing priorities

- define arrival by random priorities
- assume continuous distribution, fix.
- eg, use $2 \log n$ bits, w.h.p. no collisions

Treaps.

- tree has keys in heap order of priorities
- unique tree given priorities—follows from insertion order
- implement insert/delete etc.
- rotations to maintain heap property

Depth $d(x)$ analysis

- Tree is trace of a quicksort
- We proved $O(\log n)$ w.h.p.
- for x rank k, $E[d(x)] = H_k + H_{n-k+1} - 1$
- $S^- = \{y \in S \mid y \leq x\}$
- $Q_x =$ ancestors of x

- Show $E[Q_x^-] = H_k$.
- to show: $y \in Q_x^-$ iff inserted before all z, $y < z \leq x$.
- deduce: item j away has prob $1/j$. Add.
- Suppose $y \in Q_x^-$.
  - The inserted before x
  - Suppose some z between inserted before y
  - Then y in left subtree of z, x in right, so not ancestor
  - Thus, y before every z
- Suppose y first
  - then x follows y on all comparisons (no z splits
  - So ends up in subtree of y

Rotation analysis

- Insert/Delete time
  - define spines
  - equal left spine of right sub plus right spine of left sub
  - proof: when rotate up, on spine increments, other stays fixed.
- R_x length of right spine of left subtree
- $E[R_x] = 1 - 1/k$ if rank k
- To show: $y \in R_x$ iff
  - inserted after x
  - all z, $y < z < x$, arrive after y.
  - if z before y, then y goes left, so not on spine
- deduce: if r elts between, $r!$ of $(r+2)!$ permutations work.
- So probability $1/r^2$.
- Expectation $\sum 1/(1 \cdot 2) + 1/(2 \cdot 3) + \cdots = 1 - 1/k$
- subtle: do analysis only on elements inserted in real-time before x, but now assume they arrive in random order in virtual priorities.

skip lists

- ruler intuition
- achieve with geometric variables
- backwards analysis of search path
- insert/delete time

Shortest Paths

classical shortest paths.

- dijkstra's algorithm
- floyd's algorithm. similarity to matrix multiplication

Matrices

- length 2 paths by squaring
- matrix multiplication. strassen.
- shortest paths by "funny multiplication."
  - huge integer implementation
  - base-$(n+1)$ integers

Boolean matrix multiplication

- easy.
- gives objects at distance 2.
- gives n-mul algorithm for problem
- what about recursive?
- well can get to within 2: let T_k be boolean "distance less than or equal to 2^k. Squaring gives T_{k+1}.
- what about exact?

Seidel's distance algorithm.

- log-size integers:
  - parities suffice:
    - square G to get adjacency A', distance D'
      - if D_{ij} even then $D_{ij} = 2D'_{ij}$

- · if D_{ij} odd then $D_{ij} = 2D'_{ij} - 1$

- For neighbors i, k,
  - $D_{ij} - 1 \leq D_{kj} \leq D_{ij} + 1$
  - exists k, $D_{kj} = D_{ij} - 1$
- Parities
  - If D_{ij} even, then $D'_{kj} \geq D'_{ij}$ for every neighbor k
  - If D_{ij} odd, then $D'_{kj} \leq D'_{ij}$ for every neighbor k, and strict for at least one
- Add
  - D_{ij} even iff $S_{ij} = \sum_k D'_{kj} \geq D_{ij} d(i)$
  - D_{ij} odd iff $\sum_k D'_{kj} < D_{ij} d(i)$
  - How determine? find $S = AD'$

To find paths: Witness product.

- easy case: unique witness
  - multiply column c by c.
  - read off witness identity
- reduction to easy case:
  - suppose r columns have witness, where $2^k \leq r \leq 2^{k+1}$
  - choose each column with probability 2^{-k}.
  - prob. exactly one witness: $r \cdot 2^{-k}(1 - 2^{-k})^{r-1} \geq (1/2)(1/e^2)$

Mod 3:

- Recall some neighbor distance down by one
- so compute distances mod 3.
- suppose $D_{ij} = 1 \bmod 3$
- then look for k neighbor of i such that $D_{kj} = 0 \bmod 3$
- let $D^{(s)}_{ij} = 1$ iff $D_{ij} = s \bmod 3$
- than $AD^{(s)}$ has $ij = 1$ iff a neighbor k of i has $D^{(s)}_{kj}$
- so, witness matrix mul!