Admin
Arora talk.
No class Monday.

Review
Fingerprinting:

- Universe of size $u$
- Map to random fingerprint in universe of size $v \leq u$
- probability of collision $1/v$

Freivald’s technique

- verify matrix multiplication $AB = C$
- check $ABr = Cr$ for random $r$ in $\{0, 1\}^n$
- probability of success $1/2$
- works to check any matrix identity, not just product
- useful if matrices “implicit” like $AB$
- mapping size-$n^2$ matrices to size-$n$ vectors

In general, many ways to fingerprint explicitly represented objects. But for implicit objects, different methods have different strengths and weaknesses.

We’ll fingerprint 3 ways:

- vector multiply
- number mod a random prime
- polynomial evaluation at a random point

String matching

Checksums:

- Alice and Bob have bit strings of length $n$
- Think of $n$ bit integers $a$, $b$
- take a prime number $p$, compare $a \mod p$ and $b \mod p$ with $\log p$ bits.
- trouble if $a = b \pmod{p}$. How avoid? How likely?
  - $c = a - b$ is $n$-bit integer.
– so at most $n$ prime factors.
– How many prime factors less than $k$? $\Theta(k/\ln k)$
– so take $2n^2 \log n$ limit
– number of primes about $n^2$
– So on random one, $1/n$ error prob.
– $O(\log n)$ bits to send.
– implement by add/sub, no mul or div!

How find prime?

– Well, a randomly chosen number is prime with prob. $1/\ln n$,
– so just try a few.
– How know its prime? Simple randomized test (later)

Pattern matching in strings

• $m$-bit pattern
• $n$-bit string
• work mod prime $p$ of size at most $t$
• prob. error at particular point most $m/(t/\log t)$ from above
• so pick big $t$, union bound
• implement by add/sub as shift in bits

Fingerprints by Polynomials

Good for fingerprinting “composable” data objects.

• check if $P(x)Q(x) = R(x)$
• $P$ and $Q$ of degree $n$ (means $R$ of degree at most $2n$)
• mult in $O(n \log n)$ using FFT
• evaluation at fixed point in $O(n)$ time
• Random test:
  – $S \subseteq F$
  – pick random $r \in S$
  – evaluate $P(r)Q(r) - R(r)$
  – suppose this poly not 0
– then degree $2n$, so at most $2n$ roots
– thus, prob (discover nonroot) $|S|/2n$
– so, e.g., sufficient to pick random int in $[0, 4n]$
– Note: no prime needed (but needed for $Z_p$ sometimes)

• Again, major benefit if polynomial implicitly specified.

String checksum:
• treat as degree $n$ polynomial
• eval a random $O(\log n)$ bit input,
• prob. get 0 small

Multivariate:
• $n$ variables
• degree of term: sum of vars degrees
• total degree $d$: max degree of term.
• Schwartz-Zippel: fix $S \subseteq F$ and let each $r_i$ random in $S$
\[
\Pr[Q(r_i) = 0 \mid Q \neq 0] \leq d/|S|
\]

Note: no dependence on number of vars!

Proof:
• induction. Base done.
• $Q \neq 0$. So pick some (say) $x_1$ that affects $Q$
• write $Q = \sum_{i \leq k} x_1^i Q_i(x_2, \ldots, x_n)$ with $Q_k() \neq 0$ by choice of $k$
• $Q_k$ has total degree at most $d - k$
• By induction, prob $Q_k$ evals to 0 is at most $(d - k)/|S|$
• suppose it didn’t. Then $q(x) = \sum x_1^i Q(r_2, \ldots, r_n)$ is a nonzero univariate poly.
• by base, prob. eval to 0 is $k/|S|$
• add: get $d/|S|$
• why can we add?
\[
Pr[E_1] = Pr[E_1 \cap \overline{E_2}] + Pr[E_1 \cap E_2] \\
\leq Pr[E_1 \mid \overline{E_2}] + Pr[E_2]
\]
Small problem:

• degree $n$ poly can generate huge values from small inputs.

Solution 1:

– If poly is over $\mathbb{Z}_p$, can do all math mod $p$
– Need $p$ exceeding coefficients, degree
– $p$ need not be random

Solution 2:

– Work in $\mathbb{Z}$, deduce nonzero value from schwartz-zippel
– deduce nonzero mod random $q$ (as in string matching)
– so do all computation mod random $q$
– $q$ range must exceed bits (not value) of coeff.

Perfect matching

• Define

• Edmonds matrix: variable $x_{ij}$ if edge $(u_i, v_j)$
• determinant nonzero if PM
• poly nonzero symbolically.
  – so apply Schwartz-Zippel
  – Degree is $n$
  – So number $r \in (1, \ldots, n^2)$ yields 0 with prob. $1/n$

Det may be huge!

• We picked random input $r$, knew evaled to nonzero but maybe huge number
• How big? About $n!r^n$,
• So only $O(n \log n + n \log r)$ prime divisors
• (or, a string of that many bits)
• So compute mod $p$, where $p$ is $O((n \log n + n \log r)^2)$
• only need $O(\log n + \log \log r)$ bits
Treaps

Dictionaries for ordered sets

- New Operations.
  - enumerate in order
  - successor-of, predecessor-of (even if not in set)
  - join($S, k, T$), split, paste($S, T$)

Binary tree.

- child and parent pointers
- endogenous: leaf nodes empty.
- balanced if depth $O(\log n)$
- average case.
- worst case

Tree balancing

- rotations
- implementing operations.
- red/black, AVL
- splay trees.
  - drawbacks in geometry:
  - auxiliary structure on nodes in subtree
  - 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

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.