Announcements

• Homework 4 is available
  • Three problems
  • Program – evaluate “two choice” hashing

• Thursday, Cuckoo Hashing
  • Reading + Video link
Randomness so far

• Average case QuickSelect
• MinCut Analysis
• Binary Space Partition
• Average Case for Stable Marriage
• Primality Testing

• A random world is more predictable than a deterministic one
  • Law of large numbers
Data structures

• Keeping track of stuff
• Supporting algorithms

• Sometimes they matter and sometimes they don’t

Heapsort(A, n)

H = new Heap()
for j = 1 to n-1
    Heap.Insert(A[j])
for j = 1 to n-1
    A[i] = Heap.DeleteMin()
Data structure trade offs

• Operation Time
• Space
• Accuracy
• Implementation complexity
Hashing

• Tracking information associated with keys
  • Set
  • Search tree
  • Arrays if the keys can be an index

• Key idea – map from key space $S$ to table $T$, $|S| = n$, $|T| = m$
  • Hash function $h$, store data at location $h(x)$
  • Collision if $h(x) = h(y)$ for $x \neq y$

• In practice, $O(1)$ access
Hash functions

• Start by assuming $h$ is completely random
  • Universe $U$, $|U| = d$, table size $m$
  • $\Omega$: set of all mappings from $1..d$ to $1..m$

• Lots of work in
  • Creating practical hash functions
  • Identifying weaker assumptions than “completely random”

• For some applications, “random” hash functions are important

• Useful class of hash functions, $H = \{ H_{p,a,b} \mid p$ prime, $a, b$ in $[1 .. p-1] \}$
  • $H_{p,a,b}^p(x) = (a \times x + b) \mod p$
Collision resolution (review)

• Method 1 – Chaining (Closed addressing, open hashing)
• Method 2 – Table based (Open addressing, closed hashing)

• Load factor $\alpha$, ratio of stored elements to table size
  • For Chaining, want $0.5 \leq \alpha \leq 1$
  • For Table based, need $\alpha < 1$, $\alpha \leq 0.75$ recommended

• Common approach is to increase table size (e.g., by a factor of 2) and rehash when load factor exceeds a bound
Balls and boxes

• N boxes, repeatedly assign balls to random boxes
• Coupon collecting – expected number of balls until every box is occupied
• How about if we assign K balls at random to N boxes
  • How many cells are occupied?
  • What is the expected number of balls in the first box?
  • What is the expected maximum for the number of balls assigned to any cell?

• Balls and boxes basis for the theory of hashing
N balls in N boxes
What is the maximum number of balls in any box?

• Definition w.h.p.
  • For any j, with appropriate choice of constants, probability of failure is $O(n^{-j})$
• Maximum number of balls in a box is $O(\log n / \log \log n)$
• Log n / log log n analysis
  • Compute the probability that a given bin has more k items
  • Show that this is less than $1 / k!$
  • Choose $k = c \log n / \log \log n$, so that $1/k! < 1/n^2$
  • Probability that any bin has more than k items is less than $1/n$
The Math

\[ \Pr[\text{bin}_i \text{ gets more than } k \text{ elements}] \leq \binom{n}{k} \cdot \frac{1}{n^k} \leq \frac{1}{k!} \]

By Stirling’s formula,

\[ k! \sim \sqrt{2\pi k} \left( \frac{k}{e} \right)^k \]

If we choose \( k = O\left(\frac{\log n}{\log \log n}\right) \), we can let \( \frac{1}{k!} \leq \frac{1}{n^2} \). Then

\[ \Pr[\exists \text{ a bin } \geq k \text{ balls}] \leq n \cdot \frac{1}{n^2} = \frac{1}{n} \]

So with probability larger than \( 1 - \frac{1}{n} \),

\[ \text{max load} \leq O\left(\frac{\log n}{\log \log n}\right) \]
Power of hashing twice
Load balancing

• Let $h_1$ and $h_2$ by random hash functions
• When element $x$ is inserted, it goes to the cell $h_1(x)$ or $h_2(x)$ with least number of elements elements
• Find must check cells $h_1(x)$ and $h_2(x)$

• The maximum number of elements assigned to any cell is $O(\log\log n)$ with high probability
Proof (Intuition)

- Ball has height $k$ when it is placed in a bin with $k-1$ balls
- Expect $\leq \frac{n}{2}$ bins with 2 balls
- Expect $\leq \frac{n}{2^2}$ bins with 3 balls
- Expect $\leq \frac{n}{2^4}$ bins with 4 balls
- Expect $\leq \frac{n}{2^8}$ bins with 5 balls
- Expect $\leq \frac{n}{2^{16}}$ bins with 6 balls
- Expect $\leq \frac{n}{2^{32}}$ bins with 7 balls
Tracking keys without data

• If the key domain is [1..n] a bit vector is ideal

• What if you hash into a bit vector?

• What type of errors occur
Bloom Filter

• Basic idea – k-hash functions

• Bits are set at $h_1(x), h_2(x), \ldots, h_k(x)$
• Lookup is done by reading $h_1(x), h_2(x), \ldots, h_k(x)$

• Can we get a false negative
• Can we get a false positive
Bloom Filter

- Alternative data structures: List, Hash Table
- Critical reason for using Bloom Filter – limited storage
  - Lots of data
    - Devices with limited memory (e.g., network routers)
    - Need for main memory versus going to disk
  - Don’t need to remember the actual data (in the data structure)
- Measure of interest – number of bits per data element

- Bloom filters have been left out of computer science curriculum
Bloom Filter Example (k = 3)

```
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

```
0 1 0 0 1 0 1 0 0 0 1 1 0 0 1
```

```
0 1 0 0 1 0 1 0 0 0 1 1 0 0 1
```

```
0 1 0 0 1 0 1 0 0 0 1 1 0 0 1
```
Some Bloom Filter Math

• Table size $m$, data items $n$
• After all members of $S$ have been hashed probability of specific element being zero is
• False probability rate
• Express rate as a function of probability

$$p' = \left(1 - \frac{1}{m}\right)^{kn} \approx e^{-kn/m} = p$$

$$(1 - p')^k \approx (1 - p)^k$$

$$f' = (1 - p')^k = \left(1 - \left(1 - \frac{1}{m}\right)^{kn}\right)^k$$

$$f = (1 - p)^k = \left(1 - e^{-kn/m}\right)^k$$
False positive rate vs. $k$
Find optimal with calculus

Number of bits per member $\frac{m}{n} = 8$

optimal $k = 8 \ln 2 = 5.45\ldots$

Rewrite $f$ as

$$f = \exp(\ln(1-e^{-k/n})) = \exp(k \ln(1-e^{-k/n}))$$

Let $g = k \ln(1-e^{-k/n})$

Minimizing $g$ will minimize $f = \exp(g)$

$$\frac{\partial g}{\partial k} = \ln(1-e^{-k/n}) + \frac{k}{1-e^{-k/n}} \frac{\partial (1-e^{-k/n})}{\partial k}$$

$$= \ln(1-e^{-k/n}) + \frac{k}{1-e^{-k/n}} \frac{n e^{-k/n}}{m} = -\ln(2) + \ln(2) = 0$$

if we plug $k = (m/n) \ln 2$ which is optimal

(It is in fact a global optimum)
Bloom Filter Applications

• Dictionary to detect speling mistakes
  • All good words let through, some mistakes will happen
• List of malicious URLs in browser
• List of keys needed for a database join
• Akamai web caching, avoid caching data only requested once
  • List of requests put into a Bloom filter, store data on the second request
Bloom filter deletes

• Why do Bloom filters fail for deletes?
• Counting Bloom Filters
• Each cell is a counter (4 bits considered sufficient)
• Insert, add one to each target cell
• Delete, delete one from each target cell
• Find, test if target cells non-zero

• On overflow, leave counter at maximum value
Bloom Filter Deletes (k = 3)

Insert $X_1$
Insert $X_2$
Insert $Y_1$
Delete $Y_2$
Find $Z$