CompSci 100e
Program Design and Analysis II

February 8, 2011
Prof. Rodger
Announcements

• What is due?
  – Assignment Prestidigitation due today!
  – Apt due Feb 15
  – Markov – out today - Assignment due Feb 17
    • Lab this week also on Markov
  – Exam Feb 22

• Always turn in APTs if you have any green! You can get partial credit.

• Each APT is 10 points. You can do extra APTs to make up the missing points.
Classwork 5

• Redo ClassScores with a Map

• BTW, We are reworking some of the APTs and now to run APT you cannot have in your class
  import java.io.*.
  – That means that your Classwork 2 with classscores no longer works on the APT. You have to remove the import and the methods that use it to get it to run
Analysis – Data Structures and Algorithms

• How do we compare two programs?
• Which one will run faster? Can we tell before coding?

• Two ways:
  – Run them and see which is faster – add timings
  – Use mathematics to analyze the algorithm
Quantitative Measurements of Code

• Typically measure running time (memory?)
  – Other things to measure?

• Typically change size of input/problem to validate runtime hypotheses
  – Not the data itself, but the number of data items
  – Size of string vs. number of strings in array?

• Doubling hypothesis: What effect does doubling input size have on running time?
  – Linear: time doubles, quadratic: factor of four, ...
Different measures of complexity

• Worst case
  – Gives a good upper-bound on behavior
  – Never get worse than this
  – Drawbacks?

• Average case
  – What does average mean?
  – Averaged over all inputs? Assuming uniformly distributed random data?
  – Drawbacks?

• Best case
  – Linear search, useful?
Notations for measuring complexity

• O-notation or big-Oh: what does $n$ look like as $n$ approaches infinity?

  – $O(n)$  linear,
  – $O(n^2)$  quadratic
  – $O(n^3)$  cubic
  – $O(\log n)$  logarithmic
  – $O(2^n)$  exponential

• Sedgewick/Wayne uses tilde notation $\sim n^2$ means leading term is $n$ squared
Example

```java
for (int k=0; k<n; k++) {
    for (int j=0; j<n; j++) {
        count ++;
    }
}

What is $O()$ worst case?
Example

for (int k=1; k<n; k = k * 2) {
    for (int j=0; j<m; j++) {
        count ++;
    }
}

What is $O()$ worst case?
Example

for (int k=0; k<n; k++) {
    for (int j=k; j<n; j++) {
        count ++;
    }
}

What is $O()$ worst case?
Big-Oh, O-notation: concepts & caveats

• Count how many times “simple” statements execute
  – In the body of a loop, what matters? (e.g., another loop?)
  – Assume simple statements take a second, cost a penny,…

• In real life: cache behavior, memory behavior, swapping behavior, library gotchas, things we don’t understand,…
Simplify

\[ O(4n^3 + 100n + 35) = \]

\[ O(3n^2 + 4n + 6) = \]
Worst case

• Given an array of integers in sorted order
• What is the worst case big-Oh time for:
  – Insert(x) - insert x in the array (must maintain the sorted property.
  – Delete(x) – remove x from the array
  – Contains(x) - is x in the array, T or F
  – Min(x) – return the minimum value in the array (don’t delete it)
Multiplying and adding big-Oh

• Suppose we do a linear search then do another one
  – What is the complexity?
  – If we do 100 linear searches?
  – If we do n searches on a vector of size n?

• Binary search followed by linear search?
  – What are big-Oh complexities? Sum?
  – What about 50 binary searches? What about n searches?

• What is the number of elements in the list (1,2,2,3,3,3); (1,2,2,3,3,3,4,4,4,4)?
  – What about (1,2,2, ..., n,n,...,n)?
Helpful formulae

• We always mean base 2 unless otherwise stated
  – What is log(1024)?
  – $\log(xy) \quad \log(x^y) \quad \log(2^n) \quad 2^{(\log n)}$

• Sums (also, use sigma notation when possible)
  – $1 + 2 + 4 + 8 + \ldots + 2^k =$
  – $1 + 2 + 3 + \ldots + n =$
  – $a + ar + ar^2 + \ldots + ar^{n-1} = a(r^n - 1)/(r-1) =$
    $\sum_{i=0}^{n-1} ar^i$
Helpful formulae

• We always mean base 2 unless otherwise stated
  – What is log(1024)?
  – $\log(xy) \log(x^y) \log(2^n) 2^{(\log n)}$

• $\log(x) + \log(y)$
• $y \log(x)$
• $n \log(2) = n$
• $2^{(\log n)} = n$

• Sums (also, use sigma notation when possible)
  – $1 + 2 + 4 + 8 + \ldots + 2^k = \sum_{i=0}^{k} 2^i$
    $= 2^{k+1} - 1$
  – $1 + 2 + 3 + \ldots + n = \sum_{i=1}^{n} i$
    $= \frac{n(n+1)}{2}$
  – $a + ar + ar^2 + \ldots + ar^{n-1} = \sum_{i=0}^{n-1} ar^i$
    $= a(r^n - 1)/(r-1)$
Hashing

- Storage – hash table or hash buckets (implementation could be array, map, tree, or other structure)
- Each data item has a key
- Hash function that maps key to address in the hash table \( H(\text{key}) = \text{address} \)
- Best if the “key” is near the address
- “collision” if two items map to the same address
Example Hashing

- Duke's ACM Chapter wants to be able to quickly find out info about its members. Also add, delete and update members. Doesn't need members sorted.

  267-89-5432  John Smith
  703-25-6142  Jack Adams
  319-86-2100  Betty Harris
  476-82-5142  Rose Black

- Possible Hash Function: \( H(\text{ssn}) = \text{last 3 digits} \)
- Hash Table size is 1000, range from 0 to 999
- Will there be a collision with data above?
Bucket Hashing

Buckets 0-999 (not all buckets shown)

- Buckets hold a lot of values
- Apply hash function to data. What happens?

<table>
<thead>
<tr>
<th>Hash</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>H(267-89-5432) = 432</td>
<td>142</td>
</tr>
<tr>
<td>John Smith</td>
<td>142</td>
</tr>
<tr>
<td>H(703-25-6142) = 142</td>
<td>143</td>
</tr>
<tr>
<td>Jack Adams</td>
<td>143</td>
</tr>
<tr>
<td>H(319-86-2100 )= 100</td>
<td>432</td>
</tr>
<tr>
<td>Betty Harris</td>
<td>432</td>
</tr>
<tr>
<td>H(476-82-5142) = 142</td>
<td>742</td>
</tr>
<tr>
<td>Rose Black</td>
<td>742</td>
</tr>
<tr>
<td></td>
<td>999</td>
</tr>
</tbody>
</table>
Buckets 0-999 (not all buckets shown)

- Apply hash function to data. What happens?

H(267-89-5432) = 432
John Smith

H(703-25-6142) = 142
Jack Adams

H(319-86-2100) = 100
Betty Harris

H(476-82-5142) = 142
Rose Black

John Smith

0
100
142
143
432
742
999
Buckets 0-999 (not all buckets shown)

- Apply hash function to data. What happens?

\[
H(267-89-5432) = 432 \\
John Smith \\
H(703-25-6142) = 142 \\
Jack Adams \\
H(319-86-2100) = 100 \\
Betty Harris \\
H(476-82-5142) = 142 \\
Rose Black
\]

Where does Rose Black go?
Buckets 0-999 (not all buckets shown)

- Apply hash function to data. What happens?

\[ H(267-89-5432) = 432 \]
John Smith

\[ H(703-25-6142) = 142 \]
Jack Adams

\[ H(319-86-2100 )= 100 \]
Betty Harris

\[ H(476-82-5142) = 142 \]
Rose Black

- If buckets are one array, where does Rose Black go?

John Smith
Betty Harris
Jack Adams
Rose Black

CompSci 100, Fall 2010
If instead of Buckets, one String array 0-999, Where would Rose Black go?

- Collision, must be resolved, one way – next possible bucket (slot)
- Other Collision resolution methods

<table>
<thead>
<tr>
<th>Hash Value</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>H(267-89-5432) = 432</td>
<td>John Smith</td>
</tr>
<tr>
<td>H(703-25-6142) = 142</td>
<td>Jack Adams</td>
</tr>
<tr>
<td>H(319-86-2100) = 100</td>
<td>Betty Harris</td>
</tr>
<tr>
<td>H(476-82-5142) = 142</td>
<td>Rose Black</td>
</tr>
</tbody>
</table>

---

CompSci 100, Fall 2010
Hash Functions

• Want hash function with the fewest collisions, data distributed
• Random number Generator with seed
  – $H(\text{key})$ is $\text{random(key)} \times N$
• Shift folding with 100 buckets
  – $H(123\text{-}45\text{-}6789) = (123 + 45 + 6789) \mod 100$
• Use ascii code
  – $H(\text{BANKS}) = 66 + 65 + 78 + 75 + 83 = 367$
  – Where ascii(B) = 66, etc.
• Java – hashCode()