Lecture 20: How can computation... find what you’re looking for - faster?

Motivating Exercise

Play 20 questions in pairs
- Repeat few times
  - Person A thinks of a number between 1 and 100
  - Person B guesses number with YES/NO questions
  - Record how many guesses needed
- Switch roles and play few times more

Repeat with numbers between 1 and 1000
- Handout sheet of number grid may be useful
- Cross off guesses or numbers that secret can’t be

How many guesses on average did it take you?
- What algorithm works well?

Best algorithm for searching?

BINARY SEARCH
Guess number midway between “lo” and “hi”
((a starts out at 1, hi at 1000, midway = 500)
Ask “Lower than this midway number?”
If Yes then
  - Set hi = midway - 1
  - Guess number ½ btwn lo and hi (c 250?)
ELSE
  - Set lo = midway
  - Guess number ½ btwn lo and hi (c 750?)
Repeat

Play guessing game again with 1000 numbers - should need 10 or fewer guesses!

Use Number Grid to Track Guesses
How would you implement Binary Search for Key?

Exercise Guessing Game:
What was the secret to be guessed?
Integer between 1 and 1000 partner was thinking of

Binary Search for Specified Key:
What is the secret to be guessed?
Secret is **index in List** holding key we are looking for

Different Assumptions for Linear vs. Binary Search?

Binary search assumes list is sorted!

Data organization very important
• Can you think of other organization techniques for helping people look up data?

Trade-off: Should application pay cost to sort data or not?
• Will look at sorting later…

Review: Linear Search

Algorithm checks every element in list (in order) to see if it is the one...

Variables
• Key: Input - What we are searching for
• Key Index: Output - Index where we found Key
• index: local variable

Binary Search in Scratch

Use index to skip around List efficiently

Invariant (condition always holds true)
lo <= Index of Secret key <= hi
True before loop begins
True every time after
Running Binary Search: Ex 1

Looking for key 85

Initial
Loop | Index | Item | Item > Key? | Lo | Hi
--- | --- | --- | --- | --- | ---
1 | | | | |
2 | | | | |
3 | | | | |
4 | | | | |

Running Binary Search: Ex 2

Looking for key 33

Initial
Loop | Index | Item | Item > Key? | Lo | Hi
--- | --- | --- | --- | --- | ---
1 | | | | |
2 | | | | |
3 | | | | |
4 | | | | |

Running Binary Search: Ex 3

Looking for key 34

Initial
Loop | Index | Item | Item > Key? | Lo | Hi
--- | --- | --- | --- | --- | ---
1 | | | | |
2 | | | | |
3 | | | | |
4 | | | | |

How many guesses to find Key with Linear Search?

How many loops?
N = Elements in List
- Best case (minimum)?
  - 1 loop!
- Worst case (maximum)?
  - N loops
- Average case?
  - N/2 loops
  - O(N)
  - Just like Find Max
How many Guesses Needed?

<table>
<thead>
<tr>
<th>N</th>
<th>16</th>
<th>32</th>
<th>64</th>
<th>128</th>
<th>256</th>
<th>512</th>
<th>1024</th>
<th>2048</th>
<th>4096</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>Binary</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
</tr>
</tbody>
</table>

Decision Tree for 1..16

How many questions needed to find answer between 1 and 16 (16 numbers)?

Only 4 questions for 16 numbers

Decision Tree for 1..32

How many questions needed to find answer between 1 and 32 (32 numbers)?

Double range of numbers covered

Just one more question --> 5 questions

Decision Tree for 1..64

How many questions needed to find answer between 1 and 64 (64 numbers)?

Double again range of numbers covered

6 questions
**Decision Tree for N items**

How many questions needed for N integers?

- 1 question → 2 numbers
- 2 questions → 4 numbers
- 3 questions → 8 numbers
- 4 questions → 16 numbers
- 5 questions → 32
- 6 questions → 64
- Q questions → \(2^Q\) numbers

Each level of tree corresponds to one question

How deep (or high) is tree of N integers?

- How to go from N to Q?
- \(Q = \log_2 N\)

**How many iterations?**

- Worst case:
  - Operations: \(O(\log_2 N)\)
- Modification to decision tree
  - (2 ifs per loop)
  - Stops early if item at index = key

**Announcements**

- Project 1 Draft uploaded to Website Gallery by Friday 5pm
- Exam 1 Returned Friday
- Invitation pp 55-66 and 80-88 (Searching and complexity)

**Today’s Summary**

**Today’s Topics**
- How to efficiently search for element in a List
- \(O(N)\) guesses to find using Linear Search
- \(O(\log_2 N)\) guesses to find using Binary Search (depth of tree)
  - Assumes data is sorted

**Reading:**
- Invitation pp 55-66 and 80-88 (Searching and complexity)

**Announcements**
- Exam 1 Returned Friday
- Project 1 Draft uploaded to Website Gallery by Friday 5pm
  - Comment on others by Monday at 5pm

**Binary Search in Scratch**

**Use index to skip around**

**List efficiently**

- \(O(\log_2 N)\) guesses to find using Binary Search (depth of tree)

**Today’s Summary**