CS 171: Introduction to Computer Science II

Project Workshop and Review

Li Xiong
Today

• Project workshop
• Final review
• Course evaluation
Project Workshop

• FaceSpace
  – Alex Fields and Andres Celis

• MapQuest
  – Kenty Wang
  – Jennifer Lin
  – Xiaobo Sun
Project Workshop

- Project design – class design, algorithm design, data structures, GUI design
- Project development – development, debugging, integration
- Project experience – interesting and challenging aspects, lessons learned
- Application Demo – program features, desired features/remaining work
Review

• Data structures
• Algorithms
• When to use what
Summary of Data Structures

• General purpose data structures
  – Arrays
  – Linked lists
  – Trees
  – Hash tables
• Specialized data structures
  – Stacks
  – Queues, priority queues
  – Graphs
Summary of Algorithms

• Associated with each data structure
  – Insert
  – Delete
  – Search
  – Traversal
  – Graph algorithms

• Sorting
  – Bubble, insertion, selection
  – Merge sort
  – Quick sort
Algorithm Analysis

- Big-O notation
- Common functions
  - constant, logarithm, linear, quadratic, polynomial, exponential, factorial in increasing order of growth
- Cost analysis
  - Direct methods
  - Recursive relations (specially useful for recursive algorithms)
Programming/Problem Solving Techniques

- Recursion
- Divide and conquer
- Backtracking
- Dynamic programming – memoization
When to Use What?

• General-purpose data structures
  – Arrays (unordered)
  – Ordered arrays
  – Linked list (unordered)
  – Ordered Linked list
  – Binary search tree (unbalanced)
  – Self-balancing binary tree
  – Hash tables
### When to Use What?

- **Comparison of General Purpose Structures:**

<table>
<thead>
<tr>
<th>Data Structure</th>
<th>Search</th>
<th>Insertion</th>
<th>Deletion</th>
<th>Traversal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Array</td>
<td>$O(N)$</td>
<td>$O(1)$</td>
<td>$O(N)$</td>
<td>$-$</td>
</tr>
<tr>
<td>Ordered array</td>
<td>$O(\log N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Linked list</td>
<td>$O(N)$</td>
<td>$O(1)$</td>
<td>$O(N)$</td>
<td>$-$</td>
</tr>
<tr>
<td>Ordered linked list</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Binary tree (average)</td>
<td>$O(\log N)$</td>
<td>$O(\log N)$</td>
<td>$O(\log N)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Binary tree (worst case)</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>Balanced tree (average</td>
<td>$O(\log N)$</td>
<td>$O(\log N)$</td>
<td>$O(\log N)$</td>
<td>$O(N)$</td>
</tr>
<tr>
<td>and worst case)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hash table</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>$O(1)$</td>
<td>$-$</td>
</tr>
</tbody>
</table>
When to Use What

• Speed
  – Array and Linked Lists < Trees < Hash Table for large amount of data

• Catch
  – Trees
    • Binary search tree may be imbalanced
    • Balanced trees are complex
  – Hash tables
    • A good hash function may be difficult to achieve
    • Hard to expand (dynamic resizing of the array)
    • Performance degrade when table is too full
    • Can’t store data in sorted order
Specialized data structures

• Stacks
  – Push, pop

• Queues
  – Insert, remove

• Priority queues
  – Insert, removeMin/removeMax
  – Binary heaps

• Graphs
  – List of edges, adjacency matrix, adjacency list
  – BFS and DFS traversal algorithms
  – Shortest paths for unweighted graphs (BFS)
  – Shortest path for weighted graphs (Dijkstra, A*)
Final Exam

• May 7, 4:30 – 7pm, W303

• Exam format and difficulty level are similar to mid-term, quizzes, practice exam

• It is accumulative: 1/3 before midterm, 2/3 after midterm

• Brief review guide and practice exam are posted
Parting thoughts

• You have learned a great deal!
• Beginning at the end
• Thank you for a great semester and would love to hear from you or see you in a future class!