CS W3134: Data Structures in Java
Lecture #25: The End
12/9/04
Janak J Parekh

Administrivia
- HW#6 due on Monday
- Note duetimes
- Any questions?
- Extra TA office hours planned for Monday, I’ll let you know
- No formal office hours after Monday, although I should be available for appointments
- Fill out recommendations

Agenda
- End class
- Start final review
Intractable problems

- There are graph (and other!) problems that can’t be done in any reasonable time (linear, logarithmic, polynomial) –– they’re often exponential time, e.g., \( x^n \) –– and grow way too quickly
- Considered NP-complete (Non-deterministic Polynomial)
- Insta-Ph.D.: prove \( P=NP \) (or vice-versa)
- Example: traveling salesman problem -- visit all cities exactly once, and return to starting point, taking minimum-cost path
  - Hamiltonian cycle problem
  - \( N! \) time!

Java data structures

- Collections (container) API
- Collections and maps
  - Collections: Sets, SortedSets and Lists
  - Maps: Map and SortedMap
- Implementations:
  - Sets: HashSet, TreeSet
  - Lists: ArrayList, LinkedList
  - Maps: HashMap, TreeMap
- Lots of utility methods
  - Sort, shuffle, search, findMax/findMin
- Works with generic “Object”s
- In the real world, get comfortable with these –– they work well!

Another look at data structures

<table>
<thead>
<tr>
<th></th>
<th>List</th>
<th>Stack</th>
<th>Queue/PQ</th>
<th>Set</th>
<th>Map</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrays</td>
<td>Yes</td>
<td>Yes</td>
<td>Both</td>
<td>Poorly</td>
<td>Poorly</td>
<td></td>
</tr>
<tr>
<td>Linked Lists</td>
<td>Yes</td>
<td>Yes</td>
<td>Queue</td>
<td>Poorly</td>
<td>Poorly</td>
<td></td>
</tr>
<tr>
<td>Trees</td>
<td>Poorly</td>
<td></td>
<td>BST</td>
<td>BST</td>
<td></td>
<td>Expression, Huffman</td>
</tr>
<tr>
<td>Hashing</td>
<td></td>
<td></td>
<td>Yes</td>
<td>Yes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Heaps</td>
<td>Sort</td>
<td></td>
<td>PQ</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Graphs</td>
<td></td>
<td></td>
<td>Many</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Selected algorithms

- **Sorts**
  - Comparison-based sort
    - Bubble, selection, insertion: $O(n^2)$
    - Merge, heap: $O(n \log n)$
  - Quick: Approximately $O(n \log n)$
  - Other
    - Radix: Approximately $O(n \log n)$
    - Topological: $O(V+E)$ \{ list \}; $O(V^2)$ \{ matrix \}

- **Other**

Selected graph algorithms

- Unweighted, undirected graphs
  - Search/traversal: BFS, DFS
  - Spanning tree: BFS or DFS and store edges
- Directed graphs
  - Topological sort
  - Connectivity: Warshall
- Weighted graphs
  - Spanning tree: Prim
  - Shortest path: Dijkstra (single-source), Floyd (all-source)

The Exam

- Similar to midterm, but about 50-75% longer
- What you don’t need to know
  - Shell sort
  - Red-black trees
  - 2-3-4 trees/external storage
  - Floyd’s algorithm (too hard to do on the exam)
- What you do need to know
  - Pretty much everything else
  - Remember, stuff in class – use my slides
  - Chapter 15 is a useful overview
What’s next?

- That’s pretty much it slideswise.
- What other topics do you want to review?
- Another session next week?