Lecture 17: Heaps & Heapsort

Fall 2016

Kim Bruce & Peter Mawhorter

This Week

- Quiz on Friday:
  - Binary Trees; Binary Search Trees; Heaps
- Lab: Iterators
  - Write an iterator for `CompressedTable`
- Assignment: Darwin
  - 2-week assignment
  - Program creatures that compete and infect each other

The Midterm

- Graded!
  - Your program scores will likely bring up your grade
  - There is a second midterm

`PriorityQueue`

```
public interface PriorityQueue<E extends Comparable<E>> {
  /**
   * @pre !isEmpty()
   * @return The minimum value in the queue.
   */
  public E remove();
  public E getFirst();
  public void add(E value);
  public boolean isEmpty();
  public int size();
  public void clear();
}
```

Min-Heap

- Min-Heap H is a *complete* binary tree such that:
  - H is empty, or
  - Both of the following hold:
    - The value in the root position is the smallest value in H
    - The left and right subtrees of H are also heaps.
      Equivalent to saying parent ≤ both left and right children
- Excellent for a priority queue
  - Pop elements w/lowest priority values first

Min-Heap Example on Board

Priority Queue Impls

- Regular queue kept in order or searched:
  - One of `add` or `remove` will be O(n)
- Heap representation is more efficient: O(log n) for both `add` and `remove`.
  - `add` to heap:
    - Place in next free position
    - "Percolate" up
  - `removeMin` from heap:
    - Swap root with last and "percolate" down
    - Remove/return swapped root (no damage to tree)

`VectorHeap` Code

Comparing Sorts

- Quicksort: fastest on average (O(n log n) with good constants) but worst case is O(n^2);
  takes O(log n) extra space
- Heapsort: O(n log n) average & worst cases;
  no extra space
  - Bit slower practically than quick- & mergesorts
- Mergesort: O(n log n) average & worst cases;
  O(n) extra space
  - Works even if things don't fit in memory

Binary Search Tree

- A binary tree is a binary search tree iff:
  - It is empty, or
  - The root value is greater than or equal to every node in the left subtree and less than or equal to every node in the right subtree, and both subtrees are binary search trees

Implementation

- Recursive methods:
  - `locate` (private/protected)
  - `predecessor`
    (private/protected)
  - `add`
  - `get`
  - `remove`