CSE 326: Data Structures

More Fun with Binary Heaps

James Fogarty
Spring 2009

Administrative

• HW1 is due Today

• HW2 is posted

• Project 2 is posted
Priority Queue Operations

- insert(obj)
- deleteMin(obj)
- decreaseKey(objPtr, amount)
- increaseKey(objPtr, amount)
- remove(objPtr)
- findMax()
- expandHeap()
- buildHeap(objList)
Building a Heap

12 5 11 3 10 6 9 4 8 1 7 2

Building a Heap

12 5 11 3 10 6 9 4 8 1 7 2

Building a Heap

12 5 11 3 10 6 9 4 8 1 7 2
Building a Heap

- At every point, the new item may need to percolate all the way through the heap
- Adding the items one at a time is $O(n \log n)$ in the worst case (what is the worst case?)
- Today we get clever and do it in $O(n)$
Working on Heaps

• What are the two properties of a heap?
  – Structure Property
  – Order Property

• How do we work on heaps?
  – Fix the structure
  – Fix the order

BuildHeap: Floyd’s Method

Add elements arbitrarily to form a complete tree.
Pretend it’s a heap and fix the heap-order property!
Buildheap pseudocode

```java
private void buildHeap() {
    for (int i = currentSize/2; i > 0; i--) {
        percolateDown(i);
    }
}
```

BuildHeap: Floyd’s Method
Finally…

Note they’re not the same

O(N log N)  O(N)

But that doesn’t matter, they’re both heaps
Facts about Heaps

• Observations:
  – Inserts are at least as common as deleteMins
  – Finding a child/parent index is a multiply/divide by two
  – Each percolate step looks at only two new nodes
    • Operations jump widely through the heap

• Realities:
  – Division/multiplication by powers of two are equally fast
  – With huge data sets, disk accesses dominate
  – Looking at only two new pieces of data: bad for cache!

Cycles to access:

CPU

Cache

Memory

Disk
A Solution: d-Heaps

- Each node has d children
- Still representable by array
- Good choices for d:
  - (choose a power of two for efficiency)
  - fit one set of children in a cache line
  - fit one set of children on a memory page/disk block

One More Operation

- Merge two heaps

- Add the items from one into another?
  - O(n log n)

- Start over and build it from scratch?
  - O(n)