CSE 326: Data Structures

Skew Heaps

Steve Seitz
Winter 2009

Announcements (4/11/08)

- HW 1 due now
- HW 2 out today, due next Friday
- Project #2 Phase A out now
  - Partner sign-ups by 11:59 pm today

Merge

- Useful operation for priority queues
- Simplifies heap implementation
  - Implement other ops in terms of merge

  insert: create pq with one elt., merge
deleteMin: remove root node, merge two subtrees

How to Merge Two Binary Heaps?

1. Concat two arrays, run Floyd: \( O(n) \)
   - intelligently?
2. Insert all vals from \( T_1 \) into \( T_2 \): avg: \( O(n) \)

\[ \Omega(n) \]
Dropping the Structure Property

Amortized Complexity

Suppose you run M times and average the running times
Does it get better over time?

Amortized complexity:
max total # steps algorithm takes, in the worst case, for M consecutive operations on inputs of size N, divided by M (i.e., divide the max total by M).

Example: if M operations take total $O(M \log N)$ time in the worst case, amortized time per operation is $O(\log N)$.

Does it get better over time?

Skew Heaps

swap left-right subtrees of $a$ before merge
Example

merge

Example

merge

Example

merge

Example

merge

Example

merge
Runtime Analysis

- All operations rely on merge $\Theta(n)$

  $\Rightarrow$ worst case complexity of all ops = $\Theta(n)$

- It is known: $M$ merges take time $\Theta(M \log n)$ in the worst case

  $\Rightarrow$ amortized complexity of all ops = $\Theta(\log n)$

Skew Heap Code

```java
SkewHeap merge(heap1, heap2) {
    case {
        heap1 == NULL: return heap2;
        heap2 == NULL: return heap1;
        heap1.findMin() <= heap2.findMin():
            temp = heap1.right;
            heap1.right = heap1.left;
            heap1.left = merge(heap2, temp);
            return heap1;
        otherwise:
            return merge(heap2, heap1);
    }
}
```