Chapter 5 Heaps

Fei Tan

Chaifetz School of Business
Saint Louis University

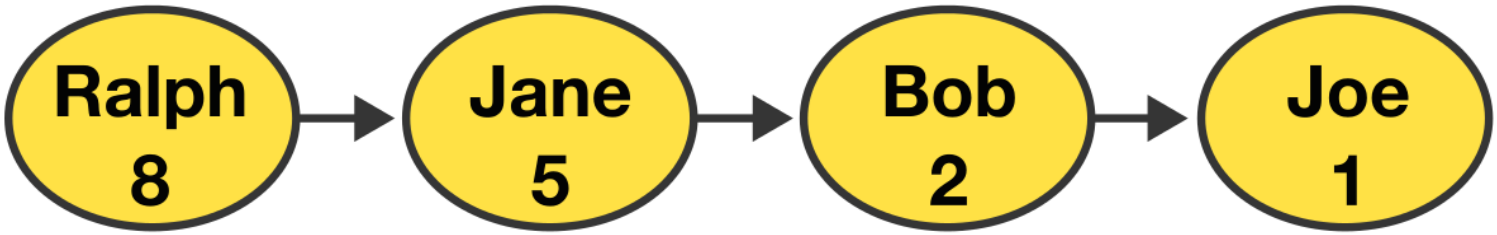
Computer Science Fundamentals
(Source: brilliant.org)

December 6, 2018

The Road Ahead...

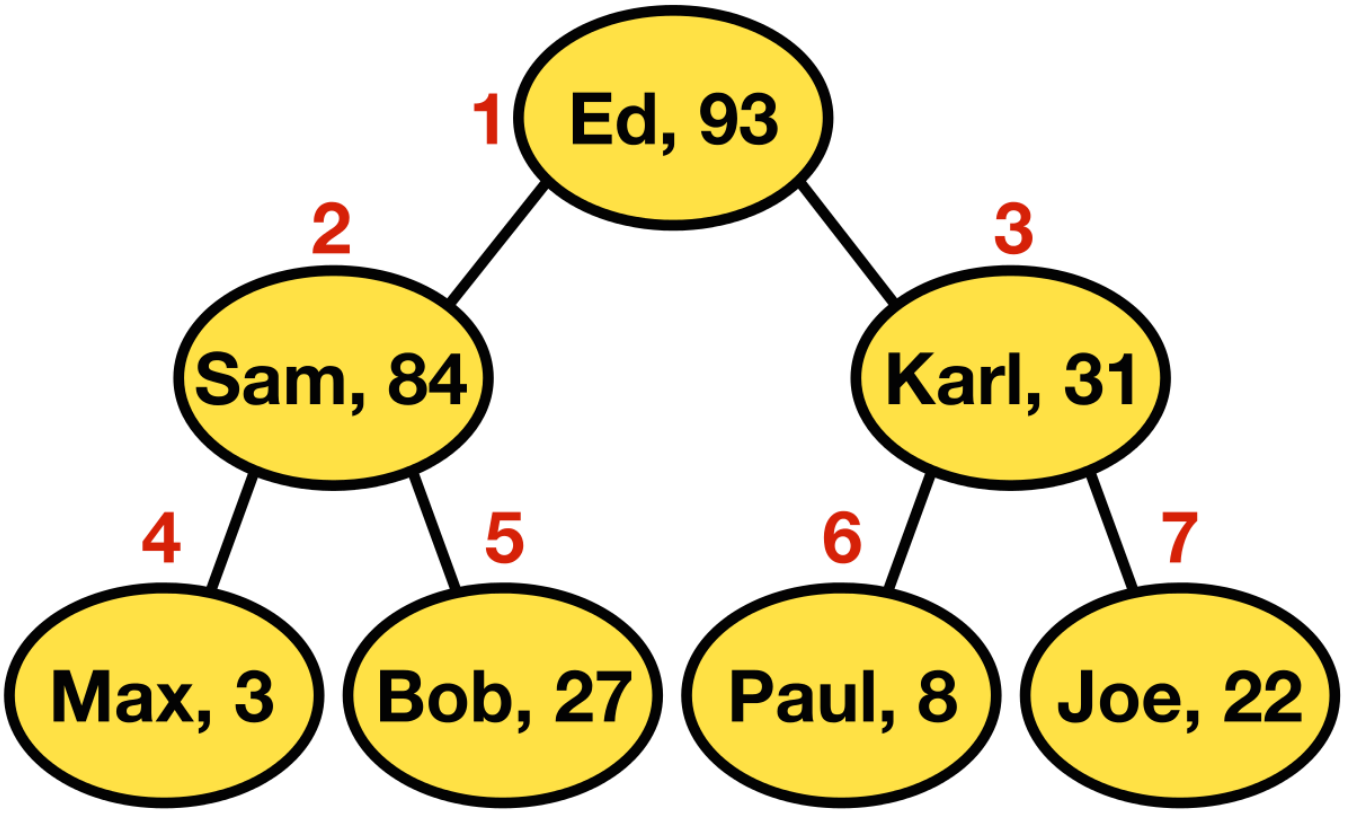
- Heaps find what we need most, without imposing any expensive requirements
- What we'll accomplish
  - how to operate priority queue
  - combine heaps & trees into treaps to balance data

Data Structure: Priority Queues



- Priority queue in linked list representation
- Basic operations:
  - insert (i, p): add element i with priority p (integer)
  - pop (): remove & return item with highest priority
  - peek (): look at value of highest priority element

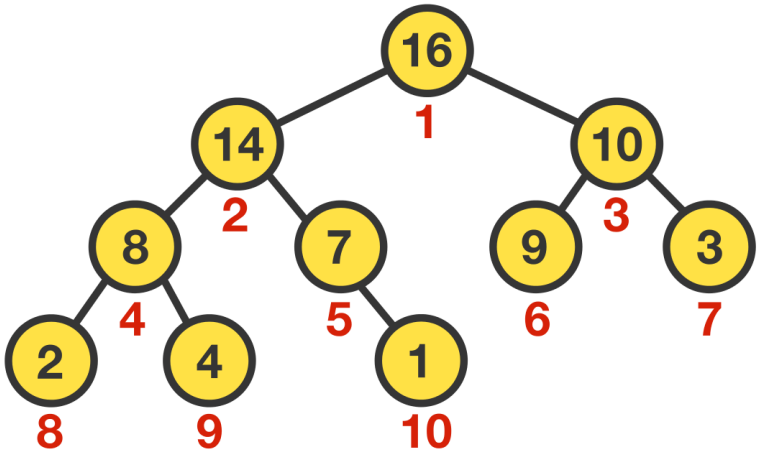
Data Structure: Binary Heaps



- Priority queue in **binary heap** representation
  - complete binary tree
  - max-heap: parent key $\geq$ children keys (black)
  - node n: left child $2n$, right child $2n + 1$ (red)

Algorithm: Heapsort

A

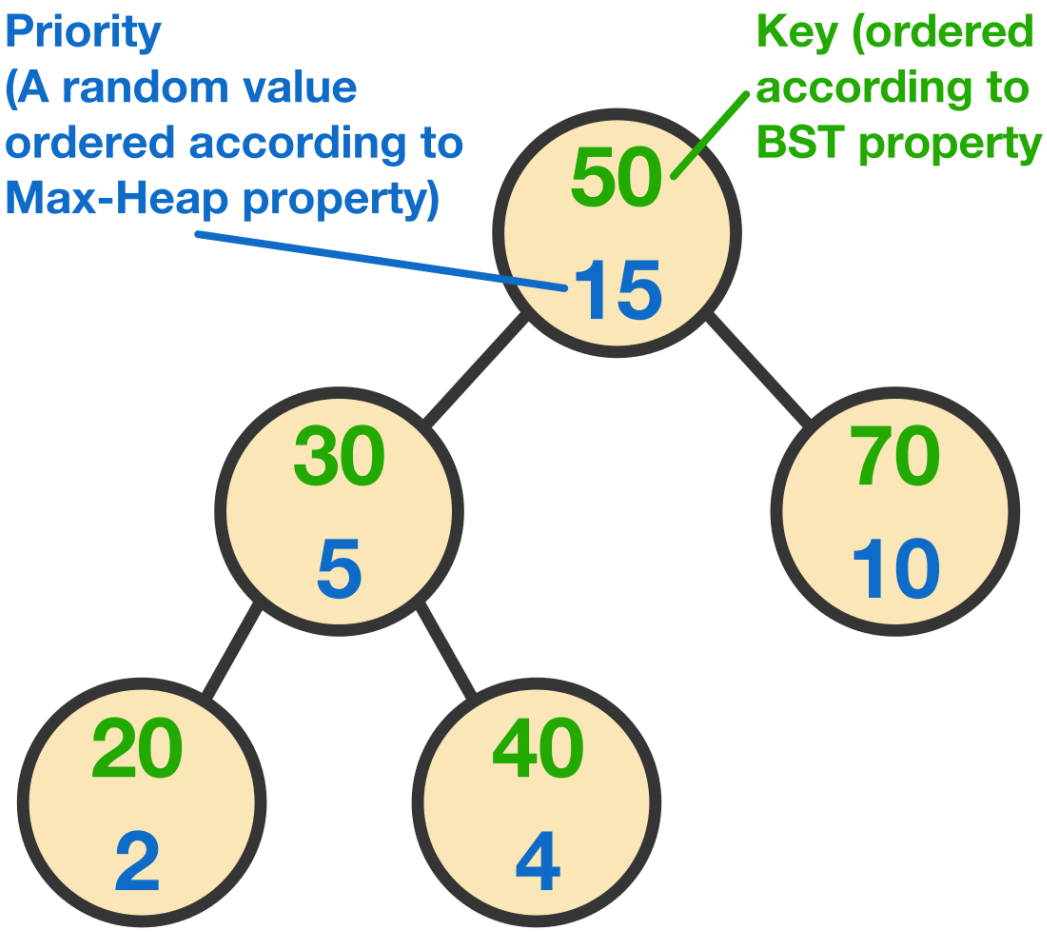


B

1	2	3	4	5	6	7	8	9	10
16	14	10	8	7	9	3	2	4	1

- Repeat following steps until empty heap
  - form binary heap by swapping children $>$ parents
  - switch root with node furthest down
  - remove root & store as smallest of sorted numbers

Data Structure: Treaps



- Treap = tree + heap
  - (priority, key) pair for each node
  - highly likely to produce well-balanced trees