6.006 Lecture 9: Sorting II

- Heapsort, heaps, priority queues 4.6
- Decision tree lower bounds 8.1
  (not covered in the lecture on Thursday)

Heapsort (+ some review of heaps)

- Arrange array elements as a heap (we didn’t see how yet)

  \[
  \begin{array}{c}
  16 \ 9 \ 10 \ \ldots \ \ldots \ \ldots \ \ldots \ 16 \\
  \end{array}
  \]

  largest \rightarrow \text{heap}

- Extract-Min and put at end; exchange max with last leaf

  \[
  \begin{array}{c}
  4 \ 11 \ 10 \ \ldots \ \ldots \ \ldots \ 11 \ 16 \\
  \end{array}
  \]

  \text{heap} \rightarrow \text{sorted}

- Heapify (fix heap)

  \[
  \begin{array}{c}
  14 \ 11 \ 10 \ 4 \ 7 \ \ldots \ \ldots \ \ldots \ 16 \\
  \end{array}
  \]
- The heap area grows and the sorted prefix shrinks after building the heap.
  
  \[
  \text{heap} \quad \rightarrow \quad \text{sorted}\]

  after 1 extract-min heapify

  \[
  \text{heap} \quad \rightarrow \quad \text{sorted}\]

  \[
  \text{heap} \quad \rightarrow \quad \text{sorted}\]

  \[
  \text{sorted, done!}\]

How to build a heap:
- fill in the array in arbitrary order

\[
5 \ 2 \ 4 \ 6 \ 1 \ 3
\]

- clearly not a heap...
- But the leaves are roots of heaps
- Let's fix the parents of the leaves; run heapify

\[
\begin{align*}
5 & \rightarrow 6 \\
2 & \rightarrow 3
\end{align*}
\]
- How long does it take to build a heap?
  
  A naive answer: $O(\lg n)$ per heapify $\Rightarrow O(n \lg n)$

  A more detailed analysis:

  $= \frac{n}{4}$ parents of leaves, 2 comparisons each $\Rightarrow \frac{n}{2}$

  $\frac{n}{8}$ grandparents, 3 comparisons or less $\Rightarrow \frac{3}{8}n$

  $\frac{1}{16}n = \frac{n}{4}$

  \[
  \sum_{k=0}^{\infty} \frac{1}{2^k} = 2 \Rightarrow \text{Build-Heap costs } O(n)
  \]
Priority Queues

An abstract data type (many implementations) useful both in algorithms (e.g., heapsort) and directly in applications.

Insert ($x$)
Maximum()
Extract-Max()
Increase-Key($x$, $v$) $x$ points to an element in $PQ$
(can also have min versions of $PQ$)

![Diagram]

Increase-Key($x$, $v$)

swap with parent until smaller than parent.

Can we also support Decrease-Key in the heap implementation of this ADT?

Yes, with heapify?