Lower Bounds on Heap Operations and Sorting

Idea:

• One might wonder if a sequence of \( n \) heap operations can be implemented in less than \( O(n\log(n)) \) time.

• Since the operations can easily be used to sort, if sorting cannot be done in less than \( O(n\log(n)) \), then neither can the heap operations.

• Although asymptotically sorting can be done faster by employing arithmetic operations in clever ways, for practical algorithms that are based on comparing elements, we can show an \( \Omega(n\log(n)) \) lower bound.
Decision Trees

Suppose that besides input and output, a sorting program $P$ consists entirely of comparisons of elements and flow of control (all of the sorting programs we have considered thus far can be expressed this way).

For any particular value of $n$, we can "unwind" all loops and recursive calls to create a decision tree that expresses all possible ways the program can work for $n$ input elements.

For example, for $n=3$, the actions of $P$ might be represented by the following tree:
Lemma: The height of a binary tree of $L$ leaves where every non-leaf vertex has exactly two children is at least $\log_2(L)$.

Proof (by induction on $L$):

• Clearly the lemma is true for $L=1$.

• For $L>1$, at least one of the two subtrees of the root has $\geq L/2$ leaves and hence by induction has height at least $\log_2(L/2) = \log_2(L)-1$.

• Since the root increases the height by 1, the lemma follows.
(lower bounds on heap operations and sorting, continued)

**Theorem:** Any program to sort, which for any particular value of $n$ can be represented by a decision tree, runs in $\Omega(n\log(n))$ time.

**Proof:** Since there are $n!$ possible permutations of the input and depending on the input values any of these permutations could be the correct sort, the decision tree must have $n!$ leaves. Hence, by the lemma above, its height must be at least:

$$\log_2(n!) \geq \log_2\left(n(n-1)\cdots\left\lceil \frac{n}{2} \right\rceil \right) \geq \log_2\left(\left(\frac{n}{2}\right)^{\left(\frac{n}{2}\right)}\right) = \frac{1}{2} n \log_2(n) - \frac{1}{2} n = \Omega(n \log(n))$$

Or, an integral can be employed to get a better approximation:

$$\log_2(n!) = (\log_2(1) + \cdots + \log_2(n))\int_1^n \log_2(x) dx = n \log_2(n) - \frac{n-1}{\ln(2)} > n \log_2(n) - 1.44n$$