Lower bounds for Sorting
A bound on the complexity of a problem is a bound on any algorithm that solves the problem

Techniques:
- Decision Tree Model
  for comparison based algorithms (i.e. sorting)
- Adversary Argument
  for selection problems
- Reduction
  show a problem solves another problem with a known lower bound

Decision Tree Model of Computation
Outcomes of comparisons determine algorithm branching
Keys are only used in comparisons (no indexing)

If by varying keys $m$ outcomes are possible then the Decision Tree must have at least $m$ leaves

Decision Tree Model of Computation (cont)

Any comparison-based sorting algorithm requires $\Omega(n \lg n)$ comparisons in the worst-case.

Proof: Let $A$ be any comparison-based sorting algorithm.
- Let $T$ be the decision tree corresponding to $A$.
- There are $n!$ outcomes of sorting $n$ keys.
- $T$ must have at least $n!$ leaves.
- A tree with $k$ leaves has height at least $\lg k$.
- $T$ has height at least $\lg n! = \Theta(n \lg n)$.
- $A$ makes at least $\Omega(n \lg n)$ comparisons in the worst-case.

QED

Decision Tree Model of Computation (cont)

Any comparison-based sorting algorithm requires $\Omega(n \lg n)$ comparisons on average.

Proof: Let $A$ be any comparison-based sorting algorithm.
- Let $T$ be the decision tree corresponding to $A$.
- There are $n!$ outcomes of sorting $n$ keys.
- Assume all $n!$ outcomes are equally probable.
- Expected number of comparisons is:
  $$\sum_{l \in \text{leaves}} \frac{1}{n!} \cdot \text{depth}(l) = \frac{1}{n!} \cdot (\text{external path length of } T)$$
Decision Tree Model of Computation (cont)

Proof: (continued)

Let \( D(m) = \) minimum external path length of any tree with \( m \) leaves.

\[
D(m) = \min_{1 \leq k < m} [D(m-k) + D(k) + m]
\]

Show \( D(m) \geq m \lg m \) by substitution method.

\[
\sum_{l \in L(T)} \frac{1}{n!} \cdot \text{depth}(l) \geq \frac{1}{n!} \cdot (n! \lg n!) = \lg n! = \Theta(n \lg n)
\]

Expected number of comparisons is \( \Omega(n \lg n) \).

QED