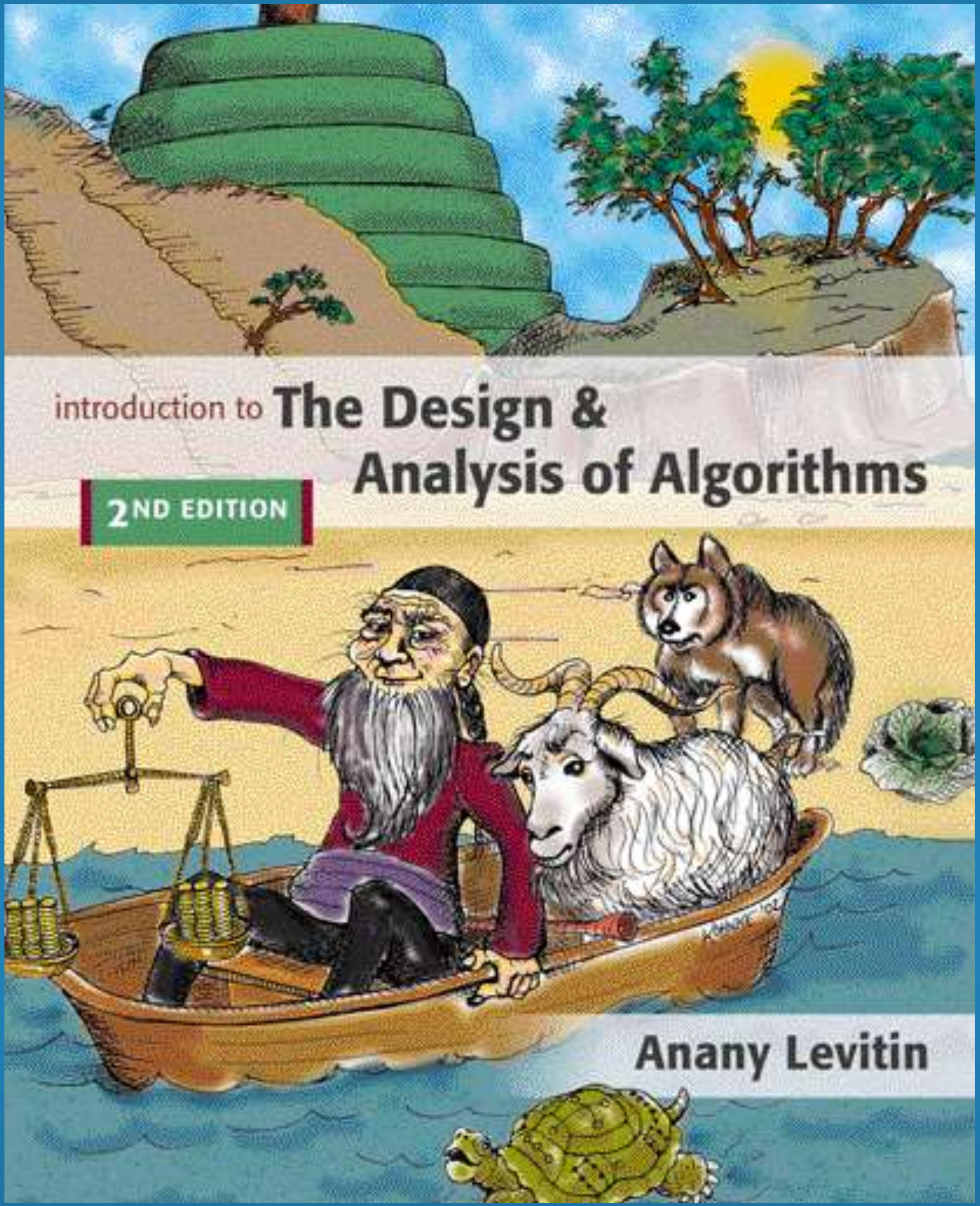
Chapter 11

Limitations of Algorithm Power

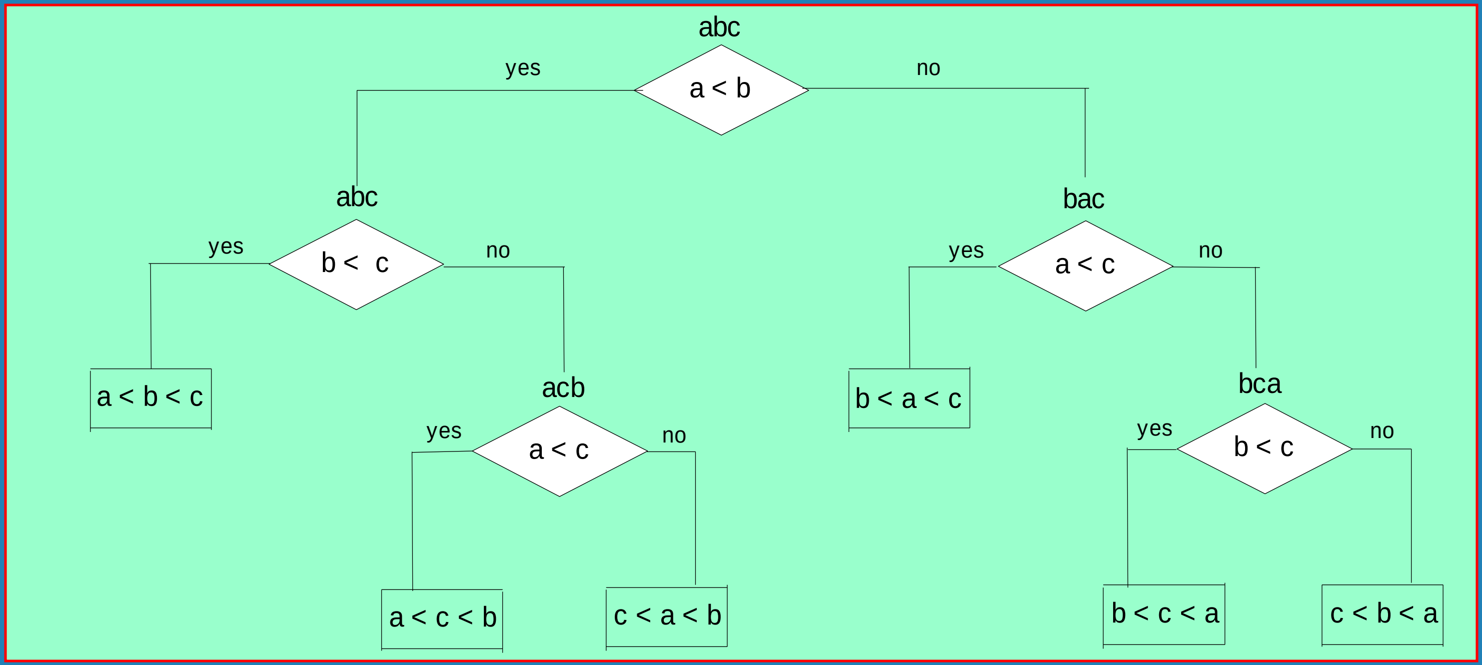


Decision Trees

Decision tree **— a convenient model of algorithms involving comparisons in which:**

- **internal nodes represent comparisons**
- **leaves represent outcomes (or input cases)**

Decision tree for 3-element insertion sort



Decision Trees and Sorting Algorithms

- **Any comparison-based sorting algorithm can be represented by a decision tree (for each fixed n)**
- **Number of leaves (outcomes) $\geq n!$**
- **Height of binary tree with $n!$ leaves $\geq \lceil \log_2 n! \rceil$**
- **Minimum number of comparisons in the worst case $\geq \lceil \log_2 n! \rceil$ for any comparison-based sorting algorithm, since the longest path represents the worst case and its length is the height**
- **$\lceil \log_2 n! \rceil \approx n \log_2 n$ (by Sterling approximation)**
- **This lower bound is tight (mergesort or heapsort)**

Ex. Prove that 5 (or 7) comparisons are necessary and sufficient for sorting 4 keys (or 5 keys, respectively).

Class *P*

P: the class of decision problems that are solvable in $O(p(n))$ time, where $p(n)$ is a polynomial of problem's input size n

Examples:

- searching
- element uniqueness
- graph connectivity
- graph acyclicity

Class *NP*

NP (***nondeterministic polynomial***): **class of decision problems whose proposed solutions can be verified in polynomial time = solvable by a *nondeterministic polynomial algorithm***

A *nondeterministic polynomial algorithm* is an abstract two-stage procedure that:

- **generates a solution of the problem (on some input) by guessing**
- **checks whether this solution is correct in polynomial time**

By definition, it solves the problem if it's capable of generating and verifying a solution on one of its tries

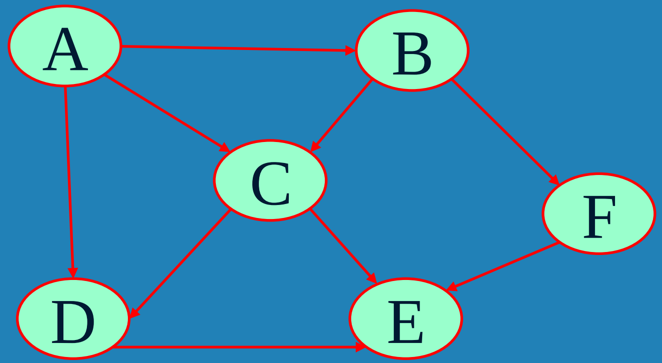
Why this definition?

- **led to development of the rich theory called "computational complexity"**

Backtracking

- **n- Queens Problem**
- n= 1 →trivial solution
- n=2 & n=3 → no solution

- **Hamiltonian Circuit Problem**
- Starts and ends with same vertex
- Visits exactly once



Continued……

- **Subset-Sum Problem**
- Find a subset of a given set S ={1,2,5,6,8} with sum d=9

Branch-and-Bound

- **Assignment Problem –(lower bound)**

	Job1	Job2	Job3	Job4
Person a	9	2	7	8
Person b	6	4	3	7
Person c	5	8	1	8
Person d	7	6	9	4

- **Knapsack Problem -(Upper bound)**

Item	Weight	Value	Value/Weight
1	4	40	10
2	7	42	6
3	5	25	5
4	3	12	4

W=10

Continued…..

- ub= v + (W-w)(vi+1/wi+1)

- Travelling Salesman Problem (lower bound)