LOWER BOUNDS & COMPUTATIONAL COMPLEXITY

CONSIDER THE SET OF ALL ALGORITHMS (KNOWN & UNKNOWN) WHICH SOLVE SOME PROBLEM P IN ALL ITS INSTANCES.

OUR GOALS ARE TWOFOLD:

1.) FIND AN ALGORITHM WHICH SOLVES P IN (WORST CASE) TIME $O(f(n))$ FOR SOME FUNCTION $f(n)$ WHICH WE AIM TO REDUCE AS FAR AS POSSIBLE.

2.) PROVE THAT ANY ALGORITHM WHICH SOLVES P MUST RUN IN (WORST CASE) TIME $\Omega(g(n))$ FOR SOME FUNCTION $g(n)$ WHICH WE AIM TO INCREASE AS FAR AS POSSIBLE.

HERE n DENOTES THE 'SIZE' OF AN INSTANCE OF PROBLEM P.

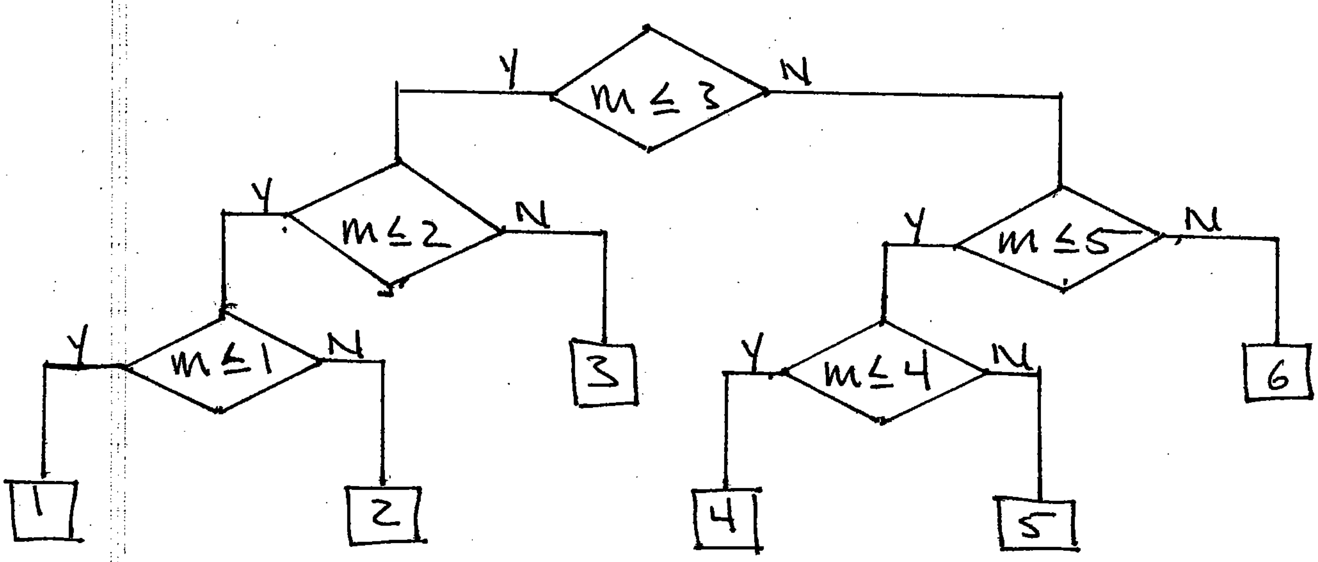
WE ARE HAPPY WHEN $f(n) = \Theta(g(n))$ FOR THEN WE KNOW WE HAVE THE BEST POSSIBLE ALGORITHM TO SOLVE P (APART FROM IMPROVEMENTS IN HIDDEN CONSTANTS.)

(1) is called <u>Algorithmics</u> by some authors, while (2) is the theory of <u>computational complexity</u>. The function $g(n)$ in (2) is called a <u>lower bound</u> on the complexity of problem P.

Decision Trees / Information Theoretic Lower Bounds

<u>Ex.</u> Let m be an integer in the range $1 \le m \le 6$. Problem: Determine the value of m by asking a sequence of yes/no questions.

This problem is similar to binary search!



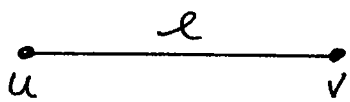
WORST CASE # COMP = 3

AVG. CASE # COMP. $= \frac{3+3+2+3+3+2}{6} = \frac{16}{6} = 2.66$

APPARENTLY THE ANSWER CAN BE OBTAINED BY ASKING NO MORE THAN 3 QUESTIONS. (will 2 suffice?)

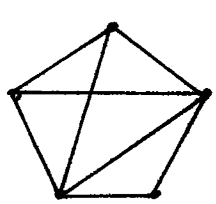
<u>REVIEW</u>: GRAPHS-TREES-ROOTED TREES.

A <u>GRAPH</u> $G = (V, E)$ is A PAIR OF SETS CALLED <u>VERTICES</u> (V) AND <u>EDGES</u> (E). EACH EDGE JOINS A (UNIQUE) PAIR OF (DISTINCT) VERTICES. TWO VERTICES WHICH ARE JOINED BY AN EDGE ARE CALLED <u>ADJACENT</u>



A <u>PATH</u> in G is A SEQUENCE OF CONSECUTIVELY ADJACENT VERTICES. G is CALLED <u>CONNECTED</u> if EVERY PAIR OF VERTICES in G ARE JOINED BY A PATH. A <u>cycle</u> is A CLOSED PATH, i.e. A PATH in WHICH THE INITIAL AND TERMINAL VERTICES ARE IDENTICAL. G is CALLED <u>ACYCLIC</u> if iT CONTAINS NO CYCLES. A <u>TREE</u> is A CONNECTED ACYCLIC GRAPH.

GRAPH:



TREE: