AS USUAL THIS STATEMENT DOES NOT ASSERT THAT THE PROBLEM CAN BE SOLVED WITH ONLY $\lceil \lg n \rceil$ COMPARISONS, ONLY THAT $\lceil \lg n \rceil$ ARE NECESSARY.

IN FACT, THE BEST KNOWN ALGORITHM DOES $n-1$ COMPARISONS, MUCH WORSE THAN $\lceil \lg n \rceil$.

FindMax(A)

```
1.) n ← length(A)
2.) max ← A[1], imax ← 1
3.) for   ← 2 TO n
4.)       if A[i] > max
5.)             max ← A[i]
6.)             imax ← i
7.) return (max, imax)
```

EX. $n = 4$

DECISION TREE : $\#\text{COMP} \geq \lceil \lg 4 \rceil = 2$

BEST KNOWN : $\#\text{COMP} = 3$

EXERCISE

DRAW A DECISION TREE FOR THE OPERATION OF FindMax IN THIS CASE. OBSERVE THAT ITS HEIGHT IS 3 NOT 2.

EX $n = 1001$

DECISION TREE : # COMP $\geq \lceil \lg 1001 \rceil = 10$

BEST KNOWN : # COMP $= 1000$

WE MUST EITHER FIND A BETTER ALGORITHM (NOT POSSIBLE) OR OBTAIN A TIGHTER LOWER BOUND

ADVERSARY ARGUMENT :

CONSIDER ANY COMPARISON BASED ALGORITHM FOR THIS PROBLEM AND LET IT RUN ON AN ARRAY $A[1 \ldots n]$, AS YET UNSPECIFIED.

DAEMON'S STRATEGY:
ANSWER EACH QUESTION CONCERNING A COMPARISON AS IF $A[i] = i \quad (1 \leq i \leq n)$. i.e. AS IF $A = (1, 2, \ldots, n)$. IN OTHER WORDS, WHEN THE ALGORITHM ASKS "IS $A[i] < A[j]$", THE DAEMON ANSWERS

$$\begin{cases} \text{TRUE} & \text{IF} \quad i < j \quad (\text{"}i \text{ HAS LOST A COMPARISON"}) \\ \text{FALSE} & \text{IF} \quad j < i \quad (\text{"}j \text{ HAS LOST A COMPARISON"}) \end{cases}$$

WHEN THIS HAPPENS WE SAY THAT THE SMALLER OF i AND j HAS "LOST A COMPARISON".

NOW ASSUME THAT THE ALGORITHM DOES FEWER THAN $M = n-1$ COMPARISONS BEFORE IT HALTS AND OUTPUTS THE INDEX k (OR THE PAIR $(A[k], k)$). I.E. THE ALGORITHM CLAIMS THAT $A[k]$ IS MAXIMUM IN ARRAY A.

LET j BE AN INTEGER SATISFYING

- $1 \leq j \leq n$
- $j \neq k$
- j HAS NOT LOST ANY COMPARISONS.

SUCH AN INTEGER MUST EXIST SINCE BY ASSUMPTION AT MOST $n-2$ COMPARISONS HAVE BEEN PERFORMED, AND EACH NEW COMPARISON CREATES AT MOST ONE NEW LOSER, HENCE THERE ARE AT MOST $n-2$ LOSERS.

AT THIS POINT THE DAEMON CAN SAY THE ALGORITHM IS WRONG BY CLAIMING THAT ARRAY A IS GIVEN BY

$$A[i] = \begin{cases} i & i \neq j \\ n+1 & i = j \end{cases}$$

INDEED $A[k] = k$ IS NOT MAXIMUM IN

THIS ARRAY, YET THE DAEMONS ANSWERS ARE ALL CONSISTENT WITH IT.

THEREFORE NO CORRECT ALGORITHM CAN SOLVE THIS PROBLEM WITH FEWER THAN $M = n-1$ COMPARISONS, AND OUR BEST KNOWN ALGORITHM CANNOT BE IMPROVED UPON.

///.

GRAPH CONNECTIVITY

LET $G = (V, E)$ BE AN (UNDIRECTED) GRAPH ON $|V| = n \geq 2$ VERTICES

PROBLEM: DETERMINE WHETHER OR NOT G IS CONNECTED.

WE CONSIDER ONLY ALGORITHMS WHICH ARE ALLOWED TO ASK QUESTIONS OF THE FORM "IS VERTEX u ADJACENT TO VERTEX v?"

THE DECISION TREE LOWER BOUND IS TRIVIAL:

OUTCOMES PER QUESTION = 2 (YES/NO)
VERDICTS = 2 (CONNECTED / DISCONNECTED.)
$\therefore h \geq \lceil \lg 2 \rceil = 1$
$\therefore$ AT LEAST 1 QUESTION IS NECESSARY.

DEPTH FIRST SEARCH (DFS) SOLVES THIS PROBLEM IN TIME $\Omega(n^2)$. (SEE SECTION 22.3 FOR A DESCRIPTION.)

ADVERSARY LOWER BOUND:

CONSIDER ANY "ADJACENCY" BASED ALGORITHM FOR THIS PROBLEM AND START IT ON AN (UNSPECIFIED) GRAPH $G=(V,E)$ WITH $n=|V|$.

DAEMON'S STRATEGY:
PARTITION V INTO TWO SUBSETS X AND Y OF SIZES $\lfloor n/2 \rfloor$ AND $\lceil n/2 \rceil$ RESPECTIVELY. I.E.

$$X \cup Y = V,\ X \cap Y = \emptyset,\ |X| = \lfloor \tfrac{n}{2} \rfloor,\ |Y| = \lceil \tfrac{n}{2} \rceil.$$

WHENEVER THE ALGORITHM ASKS "is u ADJACENT TO v?" THE DAEMON ANSWERS YES IFF u AND v BELONG TO THE SAME SUBSET. I.E.

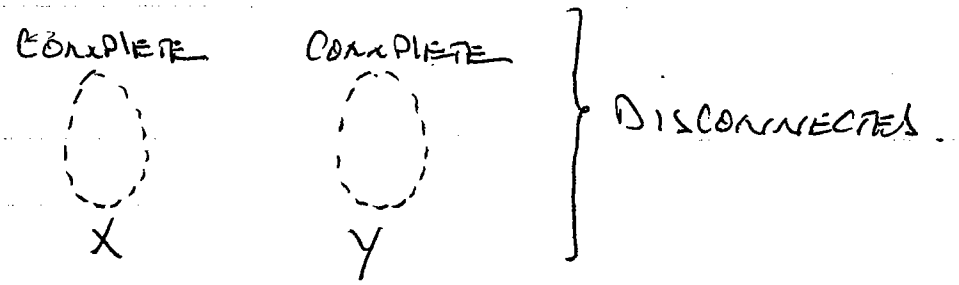
$$\begin{cases} \text{YES} & \text{IF } u,v \in X \text{ OR } u,v \in Y \\ \text{NO} & \text{IF } u \in X, v \in Y \text{ OR } u \in Y, v \in X. \end{cases}$$

IN OTHER WORDS, THE DAEMON ANSWERS AS IF G CONSISTS OF THE DISJOINT UNION OF TWO COMPLETE GRAPHS ON X AND Y RESPECTIVELY.

(A GRAPH IS CALLED <u>COMPLETE</u> IF EACH PAIR OF DISTINCT VERTICES ARE JOINED BY EXACTLY ONE EDGE.)

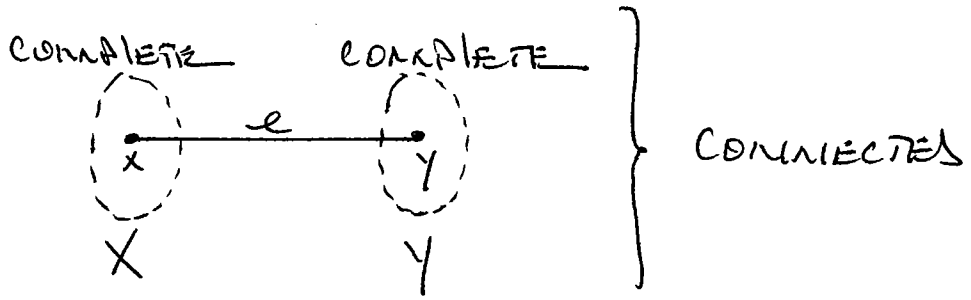
NOW SUPPOSE THE ALGORITHM HALTS AND RETURNS AN ANSWER (CONNECTED OR DISCONNECTED) AFTER ASKING FEWER THAN $M = \lfloor n/2 \rfloor \cdot \lceil n/2 \rceil$ QUESTIONS. THERE MUST THEN EXIST A PAIR $x \in X$ AND $y \in Y$ ABOUT WHICH THE ALGORITHM HAS NOT INQUIRED.

IF THE ALGORITHM SAYS G IS CONNECTED, THE DAEMON CAN CLAIM THAT G CONSISTS OF TWO DISJOINT COMPLETE GRAPHS ON X AND Y.



THIS GRAPH IS DISCONNECTED AND IS CLEARLY CONSISTENT WITH THE DAEMONS SEQUENCE OF ANSWERS.

ON THE OTHER HAND, IF THE ALGORITHM SAYS G is DISCONNECTED, THE DAEMON CAN CLAIM THAT G ACTUALLY CONSISTS OF TWO COMPLETE GRAPHS ON X AND Y, WITH A SINGLE EDGE $e = xy$ ADDED.



THIS GRAPH is CONNECTED AND is CONSISTENT WITH ALL THE DAEMON'S ANSWERS, SINCE THE EDGE $e = xy$ WAS NOT PROBED BY THE ALGORITHM.

IN EITHER CASE THE DAEMON CAN CLAIM THAT ALGORITHM is WRONG. THUS ANY ALGORITHM WHICH DOES NOT ASK AT LEAST $\lfloor n/2 \rfloor \lceil n/2 \rceil = \Omega(n^2)$ QUESTIONS (IN WORST CASE) CANNOT BE CORRECT.

///

THUS DFS CANNOT BE IMPROVED UPON, EXCEPT PERHAPS FOR IMPROVEMENTS IN HIDDEN CONSTANTS.

NOTE: A COMPLETE GRAPH ON n VERTICES HAS $\binom{n}{2} = \frac{n(n-1)}{2}$ EDGES, SINCE EACH EDGE CORRESPONDS TO A UNIQUE 2-ELEMENT SUBSET OF $V(G)$.

EXERCISE:
GIVE A MORE SOPHISTICATED ADVERSARY ARGUMENT SHOWING THAT ANY "ADJACENCY BASED" ALGORITHM TO DETERMINE CONNECTEDNESS MUST ASK AT LEAST $\binom{n}{2}$ QUESTIONS (IN WORST CASE.) NOTE: $\binom{n}{2} = \lfloor n/2 \rfloor \cdot \lceil n/2 \rceil$

IN OTHER WORDS, A CORRECT ALGORITHM MUST INQUIRE ABOUT EACH OF THE $\binom{n}{2}$ POTENTIAL EDGES.

EXERCISE
USE AN ADVERSARY ARGUMENT TO SHOW THAT $\binom{n}{2}$ "ADJACENCY" QUESTIONS ARE NECESSARY (IN WORST CASE) TO DETERMINE IF A GRAPH G is ACYCLIC.

EXERCISE
GIVE AN ADVERSARY ARGUMENT SHOWING THAT A COMPARISON SORT MUST DO AT LEAST $\lceil \lg n! \rceil$ COMPARISONS IN WORST CASE, ON INPUT OF SIZE n.

(NOTE THIS IS ESSENTIALLY NO DIFFERENT FROM THE ADVERSARY LOWER BOUND FOR 20 QUESTIONS SINCE SORTING n ELEMENTS IS REALLY A SEARCH OF $n!$ PERMUTATIONS. THE DAEMON MUST ANSWER IN A WAY WHICH KEEPS THE POOL OF CANDIDATE PERMUTATIONS AS LARGE AS POSSIBLE.)

PROBLEM

LET $b = b_1 b_2 b_3 b_4 b_5$ BE A BIT STRING OF LENGTH 5. DETERMINE WHETHER OR NOT b CONTAINS THE SUBSTRING 111 (i.e. 3 CONSECUTIVE 1's).

CONSIDER ALGORITHMS WHOSE ONLY ALLOWABLE OPERATION IS TO PEEK AT A BIT.

OBVIOUSLY 5 PEEKS ARE SUFFICIENT. A DECISION TREE ARGUMENT PROVIDES THE (USELESS) FACT THAT AT LEAST 1 PEAK IS NECESSARY

EXERCISE.

a.) USE AN ADVERSARY ARGUMENT TO SHOW THAT 4 PEEKS ARE NECESSARY IN GENERAL.

b.) DESIGN AN ALGORITHM WHICH SOLVES THE PROBLEM IN ONLY 4 PEEKS.