CSE 102 5-30-24

Recorded Lecture

<u>Problem</u> find max element in an array $A[1 \dots n]$, and where max is located.

A solution:

<u>FindMax(A, n)</u>

```
max = A[1]
imax = 1
for i = 2 to n
    if A[i] > max        basic operation
        max = A[i]
        imax = i
return (max, imax)
```

<u>Runtime</u>: $T(n) = n-1 = \Theta(n)$

Decision Tree argument:

$k = 2$, $f(n) = n$

lower bound: $\lceil \lg n \rceil$

Adversary Argument

consider any algorithm for this Problem, and run it on an array of length n.

The adversary's strategy is to answer each Probe as if $A[i] = i$ for $i = 1..n$.

i.e. as if

$$A = (1, 2, 3, \ldots, n)$$

i.e. in response to Probe: $A[i] < A[j]$, the answer is

(we say)

$$\begin{cases} \text{true} & \text{if } i < j \quad (i \text{ has lost}) \\ \text{false} & \text{if } i > j \quad (j \text{ has lost}) \end{cases}$$

Now assume the algorithm halts and returns the output

$$(A[k], k)$$

after doing fewer than $h(n) = n-1$ comparisons.

Let j be an int in range $1 \le j \le n$ such that $j \ne k$, and j has not lost any comparisons. Such an index j must exist since, by our assumption only $n-2$ comparisons have been performed, and each comparison creates at most one new loser. $\therefore$ there are at most $n-2$ losers, hence at least 2 indices have never lost a comparison.

At this point the adversary can claim

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

Note: $A[k] = k$ is not maximum in this array, $A[j] = n+1$ is maximum.

Also the adversary's seq. of answers are all consistent with this array *.

We conclude, any correct algorithm must do at least $h(n) = n-1$ comparisons to find max in an array of len. n.

$\blacksquare$

Ex.

Let $G=(V,E)$ be a graph on $|V|=n\geq 2$ vertices. Determine whether G is connected or disconnected. We consider algorithm that ask only 'adjacency' questions, or 'edge probes' i.e. "is x adjacent to y"
i.e. "does edge $\{x,y\}$ exist in E"

Decision Tree lower bound

#arity of questions $= k = 2$

#outcomes or verdicts $= f(n) = 2$

lower bound $= \lceil \lg(2) \rceil = 1$

Adversary Argument

Consider any algorithm for this Problem that asks only 'adjacency' questions, and run it against an adversary simulating a graph with n vertices ($n \geq 2$).

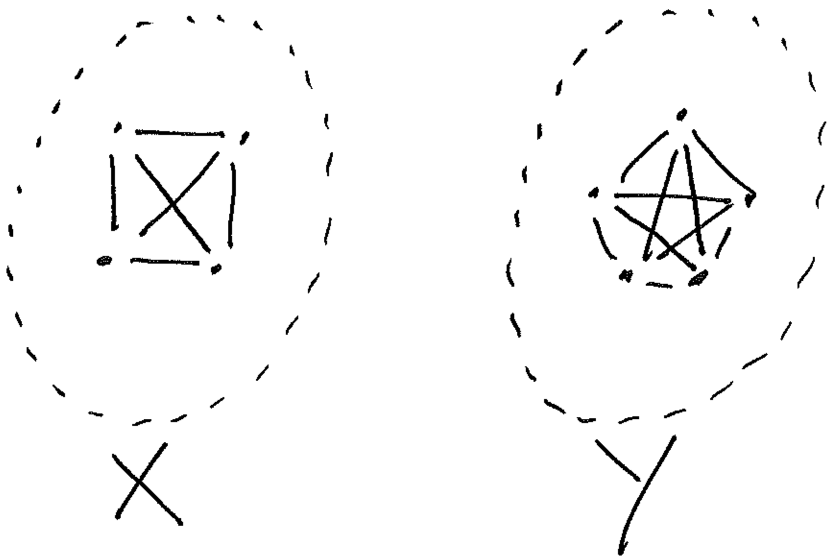
adversary strategy:

Partition V into $X, Y \subseteq V$ of sizes $|X| = \lfloor n/2 \rfloor$ and $|Y| = \lceil n/2 \rceil$. Thus $X \cup Y = V$ and $X \cap Y = \phi$.

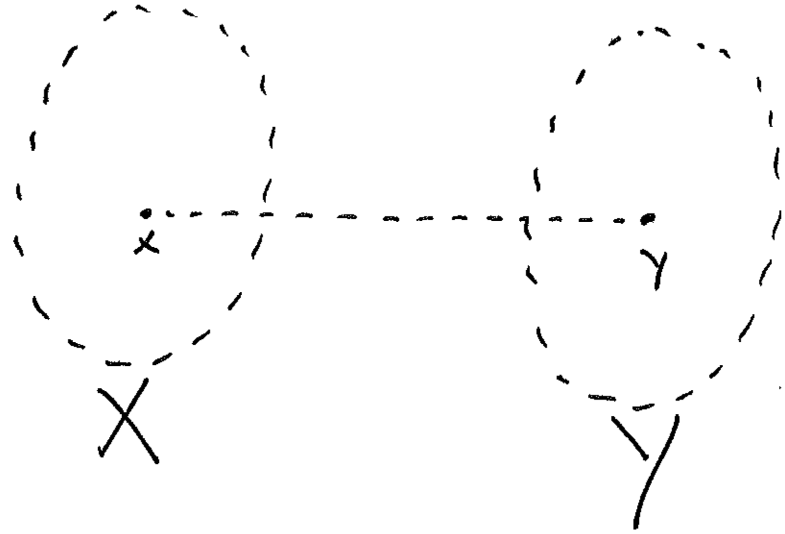
When algorithm Probes "$\{x,y\} \in E$"?, the answer given is

$$\begin{cases} \text{yes} & \text{if } x,y \in X \text{ or } x,y \in Y \\ \text{no} & \text{if } x \in X, y \in Y \text{ or } x \in Y, y \in X \end{cases}$$

i.e. adversary answers as if G consists of disjoint union of two complete $\underbrace{\text{graphs}}_{}$ sub-

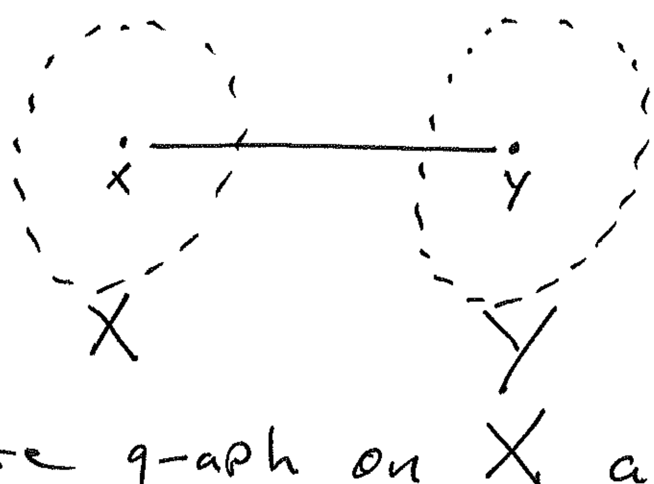


suppose algorithm halts and returns an output (connected/disconnected) after asking fewer than $h(n) = \lfloor \frac{n}{2} \rfloor \cdot \lceil \frac{n}{2} \rceil$ questions. Then there must exist vertices $x \in X$ and $y \in Y$ such that $\{x, y\}$ was not probed.



If the algorithm says G is connected, then adversary can claim G consists of 2 complete graphs on X, Y.

If algorithm says G is disconnected, adversary can claim G consists of



a complete graph on X, a complete graph on Y, and a single edge from x to y.

so any correct algorithm must do at least $h(n) = \lfloor \frac{n}{2} \rfloor \cdot \lceil \frac{n}{2} \rceil$ edge probes. ■

Remarks

- $h(n) = \lfloor \frac{n}{2} \rfloor \lceil \frac{n}{2} \rceil = \Theta(n^2)$. actually $h(n) = \frac{1}{4}n^2 + o(n^2)$, $h(n) \sim \frac{1}{4}n^2$

- DFS can solve this problem in time $\Theta(n^2)$, if we represent G as an adjacency matrix.

- note $|E(K_n)| = \binom{n}{2} = \frac{n(n-1)}{2} = \frac{1}{2}n^2 - \frac{1}{2}n$

  K_n ↑ complete graph on n vertices

K_1 K_2 K_3 K_4

Theorem

At least $\binom{n}{2}$ adjacency questions are necessary (in worst case) to determine whether a graph on n vertices is connected.

Proof ... next time ...