CSE 102 11-30-21

- Final Exam: Thur. Dec. 9
- hw7: Due Thur Dec. 2
- Final review Problems?

Continue: guessing game

adversary argument

Notation

$S_i = \{$remaining candidates for x after i^{th} question has been asked and answered.$\}$

$S_0 = \{1, 2, 3, \ldots\ldots, n\}$

note: $S_0 \supseteq S_1 \supseteq S_2 \supseteq S_3 \supseteq \ldots\ldots$

i^{th} Question : Q_i $(i = 1, 2, 3 \ldots$

$A_i(x)$ = true answer (yes/no) to Q_i if mystery number is x.

Ex. Q_1 = 'is $x \leq 50$'

$A_1(40)$ = 'yes'

$A_1(60)$ = 'no'

$$Y_i = \{ x \in S_{i-1} \mid A_i(x) = \text{yes} \}$$

$$N_i = \{ x \in S_{i-1} \mid A_i(x) = \text{no} \}$$

Ex. Q = 'is $x \leq 50$'

$$Y_1 = \{1, 2, \ldots, 50\}$$

$$N_1 = \{51, 52, \ldots, 100\}.$$

Observe $Y_i \cap N_i = \emptyset$ and $Y_i \cup N_i = S_{i-1}$

$\therefore$ $|Y_i| + |N_i| = |S_{i-1}|$

$\therefore$ at least one of Y_i or N_i must contain at least $\left\lceil \frac{|S_{i-1}|}{2} \right\rceil$ elements

<u>A's strategy</u> (A is the 'adversary' or 'daemon')

always Q_i so that S_i is the larger of Y_i or N_i. Thus

$$S_i = \begin{cases} Y_i & \text{if } |Y_i| \geq |N_i| \\ N_i & \text{if } |Y_i| < |N_i| \end{cases}$$

$\therefore$ $|S_i| \geq \left\lceil \frac{|S_{i-1}|}{2} \right\rceil$

let $b_i = |S_i|$, so $b_i \geq \left\lceil \frac{b_{i-1}}{2} \right\rceil$. Then

$$b_0 = n$$

$$b_1 \geq \left\lceil \frac{b_0}{2} \right\rceil = \left\lceil \frac{n}{2} \right\rceil$$

$$b_2 \geq \left\lceil \frac{b_1}{2} \right\rceil \geq \left\lceil \frac{\lceil n/2 \rceil}{2} \right\rceil = \left\lceil \frac{n}{2^2} \right\rceil$$

$$b_3 \geq \ldots \geq \left\lceil \frac{n}{2^3} \right\rceil$$

$$\vdots$$

$$b_i \geq \left\lceil \frac{n}{2^i} \right\rceil$$

This ends when $b_i = 1$, i.e.

$$\left\lceil \frac{n}{2^i} \right\rceil = 1$$

$$\therefore \quad 0 < \frac{n}{2^i} \leq 1$$

$$\therefore \quad n \leq 2^i$$

$$\therefore \quad \lg n \leq i$$

Since i is smallest such integer,

$$i = \lceil \lg(n) \rceil$$

<u>Exercise</u> : check that

$$\left\lceil \frac{n}{2^{\lceil \lg n \rceil}} \right\rceil = 1 \quad \text{but} \quad \left\lceil \frac{n}{2^{\lceil \lg n \rceil - 1}} \right\rceil = 2$$

Thus if B claims to know the mystery number after asking no more than $\lceil \lg n \rceil - 1$ questions, then A can claim at least one other number as his true pick, without contradicting any prior answer.

$\therefore$ the algorithm playing part of B cannot be correct.

$\therefore$ any correct algorithm for
B must ask at least $\lceil \lg n \rceil$
questions.

<u>Exercise</u>
modify above argument to get a
lower bound of $\lceil \log_k(n) \rceil$ questions
if B is allowed to ask k-ary
questions.

<u>Note:</u> these are the same
answers we got with decision
trees.

Summary of Adversary method:

1. Suppose algorithm is run against an adversary, that simulates an instance of Problem P of size n.

2. Whenever algorithm probes the data, adversary answers according to some strategy. This strategy must be specified as an algorithm, and must be consistent: there always must be at least one instance of P that would elicit the adversary's sequence of answers.

3. Prove there is a fcn $h(n)$ of input size n, with the property: If the algorithm after at most $h(n)-1$ probes, then there exists at least one instance of P (of size n) consistent with all prior answers, but whose correct solution is different from algorithm output.

4. Conclude $h(n)$ is a lower bound for worst case # of probes performed by any algorithm solving P.

__Ex.__ Given an array $A[1 \ldots n]$ of numbers, determin its maximum entry, and index where it occurs.

__FindMax (A, n)__

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

Runtime: $(n-1)$ comparisons.

__Exercise__ do a decision tree lower bound, to get: $\lceil \lg(n) \rceil$.

Exercise

draw a decision tree for above algorithm in case $n=4$. observe that its height is 3 not $\lceil \log_2 4 \rceil = 2$. (you'll have multiple leaves with same verdict.)

Adversary argument:

Given any algorithm for this Problem, run it against the following adversary that simulates an array of length n.

strategy: answer any question (comparison) as if $A = (1, 2, 3, \ldots, n)$, i.e.

$$A[i] = i \text{ for } 1 \le i \le n.$$

i.e. daemon answers "$A[i] < A[j]$?"

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

Now assume algorithm halts after doing fewer than $h(n) = n-1$ comparisons, i.e. at most $h(n)-1 = n-2$ comparisons. Suppose algorithm output is $(A[k], k)$.

Let j be an integer satisfying

$$\begin{cases} 1 \le j \le n \text{ and } j \ne k \text{ and} \\ j \text{ has not lost any comparisons} \end{cases}$$

Such a j exists since at most $n-2$ comparisons were performed, and each comp. produces at most 1 new loser.

so there are at most $n-2$ losers, hence at least $n-(n-2)=2$ indices that have lost NO comparisons. At this point, adversary claims that

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

Then this array is consistent with all the adversary's answers, but $A[k]=k$ is not maximum in this array. Such an algorithm is incorrect.

Thus any correct algorithm must do at least $h(n)=n-1$ comparisons.

Ex. Let $G = (V, E)$ be a graph with $|V| \geq 2$. $(n = |V|)$

Problem: determine whether or not G is connected.

consider algorithms that only ask questions of form: "is u adjacent to v?" we call these edge probes, or adjacency questions.

i.e. ask: $\{u,v\} \in E$ or $\{u,v\} \notin E$.

Decision tree lower bound:

k = # outcomes per question $= 2$

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

lower bound: $\lceil \log_2 2 \rceil = 1$

i.e. at least 1 adjacency question must be asked.