CSE 201 3-9-21

- SETs : if response ≥ 85%, everyone + 0.5%
- Final Exam -

Adversary arguments

Ex. Guessing game : Two players A, B.

A : chooses $x \in \{1, 2, \ldots, n\}$

B : asks $Q_1, Q_2, \ldots$ of yes/no questions.

A cheats! A must be consistent, i.e. there must always exist $x \in \{1, \ldots, n\}$ that would elicit A's seq. of answers.

A's goal : Prolong the game.

how long can A do this without being caught? The answer gives a l.b. for # of probes by any algorithm playing role of R.

must specify the algorithm that A uses.

Notation

Let S_i = {remaining candidates after Q_i is answered.}

Initially $S_0 = \{1, 2, \ldots, n\}$.

Let $A_i(x)$ = the (true) answer to Q_i if #number is x

so if $n=100$, $Q_1 =$ "is $x \le 50$", then $A_1(40) = $ 'yes', and $A_1(60) = $ 'no'.

Now define for $i \ge 1$:

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

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

Again if $n=100$, $Q_1 =$ 'is $x \le 50$', then

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

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

Adversary's strategy

answer Q_i in such a way as to imply x is in the larger of the sets Y_i, N_i

i.e. answer 'yes' if $|Y_i| \geq |N_i|$,

and 'no' if $|Y_i| < |N_i|$. Thus

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

note: $Y_i \cap N_i = \phi$, $Y_i \cup N_i = S_{i-1}$ so $|Y_i| + |N_i| = |S_{i-1}|$. Hence at least one of Y_i or N_i must contain at least $\left\lceil \frac{|S_{i-1}|}{2} \right\rceil$ elements. (Pigeonhole Principle.)

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

let $b_i = |S_i|$

so $b_0 = n$

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

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

$b_3 \geq \lceil \frac{b_2}{2} \rceil \geq \lceil \frac{\lceil n/2^2 \rceil}{2} \rceil = \lceil \frac{n}{2^3} \rceil$

$\vdots$

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

$\vdots$

when is $b_i = 1$? This implies

$\lceil \frac{n}{2^i} \rceil = 1$, i.e. $0 < \frac{n}{2^i} \leq 1$, $\therefore n \leq 2^i$

i.e. $\lg(n) \leq i$. we seek smallest such i, so $i = \lceil \lg(n) \rceil$. check

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

Thus if B stops and says he knows x after at most $\lceil \lg n \rceil - 1$ answers, then A can claim at least one other number as the true x. That other number is consistent with all answers given.

∴ $\lceil \lg n \rceil$ is a lower bound on # of questions any algorithm playing role of B must ask.

Notice: This is same lower bound who obtained by decision tree.

Exercise

modify argument to work on algorithms that ask k-ary questions: lower bound $= \lceil \log_k(n) \rceil$.

Summary

(1) suppose an algorithm solving ↑ is run against an adversary (daemon).

(2) whenever algorithm probes input data, the daemon answers according to its strategy, which must be specified. The seq. of answers must be consistent.

(3) Prove there is a fcn $h(n)$ (where n is size of input instance) with the Property: If algorithm halts after fewer than $h(n)$ probes, there exists at least one instance of $\mathcal{P}$ (of size n) consistent with all answers, but is different from algorithm's answer.

(4) conclude $h(n)$ is a L.B. for # Probes Performed by any correct algorithm.

Ex. Problem: given $A[1 \ldots n]$ of integers, find its max and the position (index) of max:

$(max, imax)$

FindMax(A, n)

1. $max = A[1]$
2. $imax = 1$
3. for $i = 2$ to n
4. if $A[i] > max$
5. $max = A[i]$
6. $imax = i$
7. return $(max, imax)$.

(# comparisons) $= n-1$.

Can we do better than $(n-1)$?

Exercise: show decision tree L.B. is $\lceil \lg n \rceil$.

Adversary strategy

Answer each probe (comparison questions like "is $A_i < A_j$") as if

$$A[i] = i$$

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

i.e. when algorithm asks: $A[i] < A[j]$?

answer { yes if $i < j$ (i has 'lost' a comp.)
no if $j < i$ (j has 'lost' a comp.)

Suppose algorithm halts after fewer than $h(n) = n-1$ comparisons, giving output: $(A[k], k)$.

Let j be an integer $1 \le j \le n$ such that (1) $j \neq k$, and (2) j has not lost any comparisons.

Note such a j must exist since at most $n-2$ comparisons have been performed, and each comparison produces at most 1 new loser. So # of losers is at most $n-2$. Hence # of non-losers is at least 2.

Now adversary claims

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

Indeed $A[k] = k$ is not maximum in this array. $A[i] = n+1$ is maximum.

All answers are consistent with this array. This no correct algorith can do fewer than $n-1$ comparisons.

So our best known algorithm is the best!

Ex. Let $G = (V, E)$ be a graph with $|V| = n \geq 2$. Problem: determine whether G is connected.

We consider algorithms that ask questions of the form

"is u adj to v ?"

We call these adjacency questions, or edge probes.

Decision Tree L.B

#verdicts = 2, (outcomes per probe) = 2

L.B. $\lceil \lg 2 \rceil = \lceil 1 \rceil = 1$.