Announcements

- **HW10** due Friday (Solving hard problems using integer programming and algorithms with huge numbers used for crypto)
- Last homework: **HW11** on computability will be due May 1st

Section plans: week of April 27-28: quiz + computability practice

week of May 4-5: quiz + review of material since the prelim

Final (cumulative): Saturday May 9th at 9am.

Thursday May 7: 4820 review led by a few TAs

- Options for last couple lectures (and other questions you may have): **see survey** (see pointer pinned on Ed)

winning option so far: divide and conquer

question: when is 6820 offered next: spring 2027

TA-ing for 4820: call will be out soon

Computability basic definitions

Focus on yes/no decision problems

Language: L = set of input with yes answer, "input accepted"

examples: - graphs that have a Hamiltonian Path

- bipartite graphs
- all Phyton programs that correctly solve one given SAT problem

Recognizable:

L recognizable if $\exists$ program that in finite time answers yes on input $x \in L$

Decidable:

L decidable if recognizable, on input $x \notin L$ algorithm output no in finite time

The Halting Problem and "co-Halting"

L_h = halting problem = { (program M, input x) such that M running on input x terminates }

does not run forever

Reminder

L_h is recognizable (just run M on x)

see Friday: not decidable

$L_{co\text{-}h}$ = { (program M, input x) such that M on x runs forever }

co-halting

Claim: L_h not decidable $\Rightarrow$ $L_{co\text{-}h}$ not recognizable

same meaning as no algorithm correctly decides $(M, x) \in L_h$

Join by Web PollEv.com/evatardos772

Is the co-halting problem decidable or recognizable?

A. co-halting problem is decidable
B. co-halting problem is recognizable, but not decidable
C. co-halting problem is not even recognizable

total
= programs M
so that it terminates
on all inputs

Reduction to prove hardness

- Proving L not decidable

  $L_H \leq L$ prove this

  if L can be decided then can use deciding algorithm to decide the halting problem

- Proving L not recognizable

  $L_{\text{co-halt}} \leq L$

reduction typically in poly time, but this is not required needs to be finite

Accept problem

L_{Accept} = { (program M, input x), running M on x results in answer "yes" in finite time }

Recognizable — simply run M on x & accept if you get "yes"

Claim: $L_{halt} \leq L_{accept}$ (⇒ Accept is not decidable)

Proof: input (M, x) to halting

write wrapper M' — run M on x
if terminates (ignore output)
return yes

$(M, x) \in L_{halt}$ if & only if $(M', x) \in L_{accept}$

Totality

$L_T = \{$ program M; halts on all inputs $x\}$

Claim $L_{halt} \leq L_T \Rightarrow$ not decidable

Proof input (M,x) halting $\longrightarrow$

generate M' ignores input runs M on x

$(M,x) \in L_{halt} \iff M' \in L_T$

Claim $L_{co\text{-}halt} \leq L_T \Rightarrow$ not recognizable

Proof: input (M,x) co-halting

$(M,x) \in$ co-halt $\iff M'' \in L_T$

M'' on input y

run M on x only for $|y|$ steps

If M terminates

enter infinite loop

else terminate

Busy beaver

$B(n)$ = most instructions a Python program can do with n-character program (not infinite loop)

Claim $B(n)$ not computable

Proof idea: $\alpha_{halt} \leq$ computing $B(n)$

input (M, x) to halting

check length of program $M = n$

run M in x for $B(n)$ steps

if does not terminate, return no