Other Undecidable Problems

Strategy for more problems:

1. *Assume for purpose of contradiction that problem P decid.*
2. *Show that if P were decidable, another problem Q (that we have shown undecidable) would be decidable. CONTR.*

Th. E_{TM} = {<M> | M is a TM and L(M) is empty} is undecidable.

Proof idea: We show if E_{TM} were decidable, then A_{TM} would be.

So suppose E_{TM} decided by R. We want decider S for A_{TM}.

Could we use R directly for S? R accepts <M> iff L(M) = ϕ
So if L(M) = ϕ, then S should reject w.
But if L(M) not = ϕ, then we're stuck!!????

Other Undecidable Problems

Trick: Instead of running R on <M>, we run it on a different machine $<S^{M,w}>$ (defined for specific M and w).

$S^{M,w}$ takes input x, and :

1. *if x not = w, it rejects.*
2. *if x = w, run M on input w, and accept if M does.*

$S^{M,w}$ accepts at most one input, w; and it accepts w iff M accepts w.

so $L(S^{M,w}) = \phi$ iff M rejects w
so $L(S^{M,w}) \neq \phi$ iff M accepts w

This allows us to use problem E_{TM} !

E_{TM} is Undecidable

Proof: Suppose E_{TM} were decidable by R. We show that A_{TM} would be decidable, a contradiction.

We first define $S^{M,w}$ for given M and w as:

On input x

1. if x ≠ w, reject.

2. if x = w, run M on w and accept if M does.

We now define decider S for A_{TM} as follows:

On input <M,w>

1. use M to construct $S^{M,w}$

2. Run R on <$S^{M,w}$>

3. If R accepts, S rejects; if R rejects, S accepts.

Other Undecidable Problems

We can use other problems than A_{TM} to get a contradiction!

Th. EQ_{TM} = {<M1,M2> | M1 is a TM , L(M1) = L(M2)} is undecidable.

Proof: We show if EQ_{TM} is decidable, E_{TM} is, Contr.

Proof idea: EQ_{TM} tests if 2 languages =, E_{TM} if language = ϕ

So E_{TM} is a special case of EQ_{TM}.

Proof: Suppose R decides EQ_{TM}. Construct S to decide E_{TM} as follows.

S: on input <M>

1. Run R on <M, Rej>, where Rej is TM with L(Rej)= ϕ

2. If R accepts, S accepts; if R rejects, S rejects.

CONTR. !

Mapping Reducibility

Intuition: to solve a problem, reduce it to another

Ex. to solve problem of getting A in course, reduce to getting A on quizzes, on homework, and on final.

We've already used this idea to prove more problems undecidable

Ex : If $HALT_{TM}$ is decidable, then A_{TM} is decidable.

But A_{TM} already shown undecidable, CONTR.

Mapping reduciblity makes reduction of problems precise

$A \leq_m B$: intuitively, if B has a solution, can use to solve for A.

We will use computable mapping m of A to B

5

Computable Functions

TM as computer of output, not recognizer

Output is what is on tape when halt

Def.: $f : \Sigma^* \longrightarrow \Sigma^*$ is a *computable* **function if there is some TM M which computes f:**
on input w, M halts with just f(w) on its tape.

Note: M must always halt!

Ex. HW 3 problem 6: TM that, with i and j on tape, computes i*j (if on just 1 tape, and erased i and j, then this TM would compute * function)

Other computable functions: factorial(<n>)

gcd(<m,n>) prime(<n>) = <nth prime no>

Tranformers of TM:
f(<M>) = <M'> where L(M) = L(M'), and M' never moves its tape off the left end

6

Mapping Reducibility

Def. Language A is *mapping reducible* **to language B, written** $A \leq_m B$**, if there is a computable function**

f: $\Sigma^* \rightarrow \Sigma^*$**, where for every w in** Σ^***,**

w in A $\longleftrightarrow$ **f(w) in B**

f is called a reduction

Can test membership in A by membership in B

Ex. $E_{TM} \leq_m EQ_{TM}$**.**

f: $\Sigma^* \rightarrow \Sigma^*$

<M> $\rightarrow$ **f(<M>) = <M,Rej>** *(where L(Rej) =* ϕ *)*

f appends to <M> the repr. <Rej>; f(w) = w ow.

<M> in E_{TM} $\longleftrightarrow$ **f(<M>) in** EQ_{TM}

f is computable:

7

Uses of Mapping Reducibility

Th: If $A \leq_m B$ **and B is decidable, then A is decid.**

Proof: Let D be the decider for B, and f the reduc. A $\rightarrow$ **B. We define decider D' for A as follows:**

D': on input w

1. **compute f(w)**
2. **Run D on input f(w); if D accepts, D' accepts. if D rejects, D' rejects.**

Since w in A $\longleftrightarrow$ **f(w) in B by def. map. reduc., and D' accepts w** $\longleftrightarrow$ **D accepts f(w), we conclude D' accepts A.**

Since D is a decider, D' is a decider.

Therefore, A is decidable.

8

Uses of Mapping Reducibility

Cor: If $A \leq_m B$ and A is undecidable, then B is undecidable.

Proof: Suppose B were decidable. By the prev. Th., since $A \leq_m B$, then A would be decidable. Contradiction.

Ex. If $A_{TM} \leq_m HALT_{TM}$ and A_{TM} undecidable

$\longrightarrow$ $HALT_{TM}$ undecidable

To show $A_{TM} \leq_m HALT_{TM}$, need computable

f : <M,w> $\longrightarrow$ <M',w'>

<M,w> in A_{TM} iff <M',w'> in $HALT_{TM}$

9

Uses of Mapping Reducibility

TM F computes f as follows:

F: on input <M,w>

1. Construct M':

   M': on input x

   1. Run M on x.
   2. If M accepts, M' accepts.
      If M rejects, loop infinitely.

2. Output <M', w>

Then <M,w> in A_{TM} $\longleftrightarrow$ M accepts w $\longleftrightarrow$

M' halts on w $\longleftrightarrow$

<M',w> in $HALT_{TM}$

10

Uses of Mapping Reducibility

Th. E_{TM} **is undecidable** *(revisited)*

A_{TM} **reduced to** E_{TM}

Qu: is it a mapping reducibility?

<M,w> $\longrightarrow$ <M'>

M accepts w $\leftrightarrow$ L(M') $\neq \phi$

<M,w> in A_{TM} $\leftrightarrow$ <M'> in E_{TM}

so we showed $A_{TM} \leq_m \overline{E}_{TM}$

not $A_{TM} \leq_m E_{TM}$ *(can't be done)*

We could still prove the th., because

E_{TM} decidable $\leftrightarrow$ $\overline{E}_{TM}$ decidable

11

Turing-Recognizability & Mapping Reducibility

Th. If $A \leq_m B$ **and B Turing-recognizable, then A is Turing-recognizable.**

Proof: Suppose B is recognized by TM R, and f: A $\rightarrow$ B is a reduction. We define recognizer TM M for A as follows:

M: on input w

1. **compute f(w)**
2. **run R on input f(w); if R accepts, M acc. if R rejects, M rejects.**

Since w in A $\leftrightarrow$ f(w) in B, by def. map. reduc., and M accepts w $\leftrightarrow$ R accepts f(w) we conclude M accepts A. M is certainly a recognizer. Therefore, A is Turing-recognizable.

12