CS21
Decidability and Tractability

Lecture 13

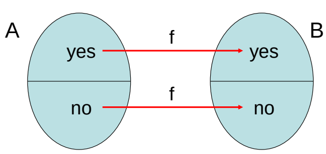
February 3, 2012

Outline

- undecidable problems
  - surprising contrasts between decidable/undecidable
- Rice's Theorem
- a non-RE and non-co-RE language

Definition of reduction

- More refined notion of reduction:
  - "many-one" reduction (commonly)
  - "mapping" reduction (book)



reduction from language A to language B

Linear Bounded Automata

LBA definition: TM that is prohibited from moving head off right side of input.

- machine prevents such a move, just like a TM prevents a move off left of tape

- How many possible configurations for a LBA M on input w with |w| = n, m states, and p = |Γ| ?
  - counting gives: mnp^n

Dec. and undec. problems

- two problems we have seen with respect to TMs, now regarding LBAs:
  - LBA acceptance:

    A_{LBA} = {<M, w> : LBA M accepts input w}
  - LBA emptiness:

    E_{LBA} = {<M> : LBA M has L(M) = Ø}
- Both decidable? both undecidable? one decidable?

Dec. and undec. problems

Theorem: A_{LBA} is decidable.

Proof:

- input <M, w> where M is a LBA
- key: only mnp^n configurations
- if M hasn't halted after this many steps, it must be looping forever.
- simulate M for mnp^n steps
- if it halts, accept or reject accordingly,
- else reject since it must be looping

Dec. and undec. problems

Theorem: E_{LBA} is undecidable.

Proof:

- reduce from co-A_{TM} (i.e. show co-$A_{TM} \leq_m E_{LBA}$)
- what should f(<M, w>) produce?
- Idea:
  - produce LBA B that accepts exactly the accepting computation histories of M on input w

Dec. and undec. problems

Proof:

- f(<M, w>) = <B> described below

on input x, check if x has form

$\#C_1\#C_2\#C_3\#...\#C_k\#$

- check that C_1 is the start configuration for M on input w
- check that $C_i \Rightarrow^1 C_{i+1}$
- check that C_k is an accepting configuration for M

- is B an LBA?
- is f computable?
- YES maps to YES?

  <M, w> $\in$ co-$A_{TM} \Rightarrow$ f(M, w) $\in E_{LBA}$
- NO maps to NO?

  <M, w> $\notin$ co-$A_{TM} \Rightarrow$ f(M, w) $\notin E_{LBA}$

Dec. and undec. problems

- two problems regarding Context-Free Grammars:
  - does a CFG generate all strings:

    ALL_{CFG} = {<G> : G is a CFG and L(G) = Σ^*}
  - CFG emptiness:

    E_{CFG} = {<G> : G is a CFG and L(G) = Ø}
- Both decidable? both undecidable? one decidable?

Dec. and undec. problems

Theorem: E_{CFG} is decidable.

Proof:

- observation: for each nonterminal A, the set

  $S_A = \{w : A \Rightarrow^* w\}$

  is non-empty iff there is some rule:

  $A \rightarrow \mathbf{x}$

  and $\forall$ non-terminals B in string x, $S_B \neq \emptyset$

Dec. and undec. problems

Proof:

- on input <G>
- mark all terminals in G
- repeat until no new non-terminals get marked:
  - if there is a production $A \rightarrow x_1x_2x_3\ldots x_k$
  - and each symbol $x_1, x_2, \ldots, x_k$ has been marked
  - then mark A
- if S marked, reject ($G \notin E_{CFG}$), else accept ($G \in E_{CFG}$).
- terminates? correct?

Dec. and undec. problems

Theorem: ALL_{CFG} is undecidable.

Proof:

- reduce from co-A_{TM} (i.e. show co-$A_{TM} \leq_m ALL_{CFG}$)
- what should f(<M, w>) produce?
- Idea:
  - produce CFG G that generates all strings that are ***not*** accepting computation histories of M on w

Dec. and undec. problems

Proof:

– build a NPDA, then convert to CFG

– want to accept strings not of this form,

$\#C_1\#C_2\#C_3\#...\#C_k\#$

plus strings of this form but where

- C_1 is not the start config. of M on input w, or
- C_k is not an accept. config. of M on input w, or
- C_i does not yield in one step C_{i+1} for some i

Dec. and undec. problems

Proof:

– our NPDA nondeterministically checks one of:

- C_1 is not the start config. of M on input w, or
- C_k is not an accept. config. of M on input w, or
- C_i does not yield in one step C_{i+1} for some i
- input has fewer than two #'s

– details of first two?

– to check third condition:

- nondeterministically guess C_i starting position
- how to check that C_i doesn't yield in 1 step C_{i+1} ?

Dec. and undec. problems

Proof:

– checking:

- C_i does not yield in one step C_{i+1} for some i

– push C_i onto stack

– at #, start popping C_i and compare to C_{i+1}

- accept if mismatch away from head location, or
- symbols around head changed in a way inconsistent with M's transition function.

– is everything described possible with NPDA?

Dec. and undec. problems

Proof:

– Problem: cannot compare C_i to C_{i+1}

– could prove in same way that proved

$\{ww: w \in \Sigma^*\}$ not context-free

– recall that

$\{ww^R: w \in \Sigma^*\}$ is context-free

– free to tweak construction of G in the reduction

– solution: write computation history:

$\#C_1\#C_2^R\#C_3\#C_4^R...\#C_k\#$

Dec. and undec. problems

Proof:

– f(<M, w>) = <G> equiv. to NPDA below:

on input x, accept if not of form:

$\#C_1\#C_2^R\#C_3\#C_4^R...\#C_k\#$

- accept if C_1 is the not the start configuration for M on input w
- accept if check that C_i does not yield in one step C_{i+1}
- accept if C_k is not an accepting configuration for M

- is f computable?
- YES maps to YES?

  $\langle M, w\rangle \in \text{co-}A_{TM} \Rightarrow f(M, w) \in ALL_{CFG}$

- NO maps to NO?

  $\langle M, w\rangle \notin \text{co-}A_{TM} \Rightarrow f(M, w) \notin ALL_{CFG}$

Rice's Theorem

- We have seen that the following properties of TM's are undecidable:
  – TM accepts string w
  – TM halts on input w
  – TM accepts the empty language
  – TM accepts a regular language
- Can we describe a single generic reduction for all these proofs?
- Yes. *Every* property of TMs undecidable!

Rice's Theorem

- A TM property is a language P for which
  - if $L(M_1) = L(M_2)$ then $<M_1> \in P$ iff $<M_2> \in P$
- TM property P is nontrivial if
  - there exists a TM M_1 for which $<M_1> \in P$, and
  - there exists a TM M_2 for which $<M_2> \notin P$.

Rice's Theorem: Every nontrivial TM property is undecidable.

Rice's Theorem

- The setup:
  - let $T_\varnothing$ be a TM for which $L(T_\varnothing) = \varnothing$
    - technicality: if $<T_\varnothing> \in P$ then work with property co-P instead of P.
    - conclude co-P undecidable; therefore P undec. due to closure under complement
  - so, WLOG, assume $<T_\varnothing> \notin P$
  - non-triviality ensures existence of TM M_1 such that $<M_1> \in P$

Rice's Theorem

Proof:
- reduce from A_{TM} (i.e. show $A_{TM} \leq_m P$)
- what should f(<M, w>) produce?
- f(<M, w>) = <M'> described below:

on input x,
• accept iff M accepts w and M_1 accepts x
(intersection of two RE languages)

- f computable?
- YES maps to YES?

  $<M, w> \in A_{TM} \Rightarrow$
  $L(f(M, w)) = L(M_1) \Rightarrow$
  $f(M, w) \in P$

Rice's Theorem

Proof:
- reduce from A_{TM} (i.e. show $A_{TM} \leq_m P$)
- what should f(<M, w>) produce?
- f(<M, w>) = <M'> described below:

on input x,
• accept iff M accepts w and M_1 accepts x
(intersection of two RE languages)

- NO maps to NO?

  $<M, w> \notin A_{TM} \Rightarrow$
  $L(f(M, w)) = L(T_\varnothing) \Rightarrow$
  $f(M, w) \notin P$

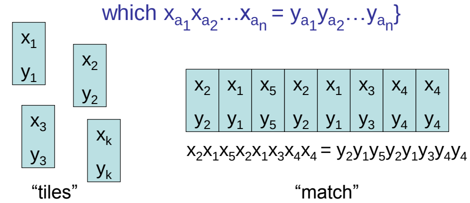
Post Correspondence Problem

- many undecidable problems unrelated to TMs and automata
- classic example: Post Correspondence Problem

$PCP = \{<(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)> :$
$x_i, y_i \in \Sigma^*$ and there exists $(a_1, a_2, \ldots, a_n)$ for
which $x_{a_1}x_{a_2}\ldots x_{a_n} = y_{a_1}y_{a_2}\ldots y_{a_n}\}$

Post Correspondence Problem

$PCP = \{<(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)> :$
$x_i, y_i \in \Sigma^*$ and there exists $(a_1, a_2, \ldots, a_n)$ for
which $x_{a_1}x_{a_2}\ldots x_{a_n} = y_{a_1}y_{a_2}\ldots y_{a_n}\}$



Post Correspondence Problem

Theorem: PCP is undecidable.

Proof:

- reduce from A_{TM} (i.e. show $A_{TM} \leq_m$ PCP)
- two step reduction makes it easier
- first, show $A_{TM} \leq_m$ MPCP

  (MPCP = "modified PCP")
- next, show MPCP $\leq_m$ PCP

Post Correspondence Problem

MPCP = $\{<(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)> :$
$x_i, y_i \in \Sigma^*$ and there exists $(a_1, a_2, \ldots, a_n)$ for which $x_1 x_{a_1} x_{a_2} \ldots x_{a_n} = y_1 y_{a_1} y_{a_2} \ldots y_{a_n}\}$

Proof of MPCP $\leq_m$ PCP:

- notation: for a string $u = u_1 u_2 u_3 \ldots u_m$
  - $*u$ means the string $*u_1*u_2*u_3*u_4\ldots*u_m$
  - $u*$ means the string $u_1*u_2*u_3*u_4\ldots*u_m*$
  - $*u*$ means the string $*u_1*u_2*u_3*u_4\ldots*u_m*$

Post Correspondence Problem

Proof of MPCP $\leq_m$ PCP:

- given an instance $(x_1, y_1), \ldots, (x_k, y_k)$ of MPCP
- produce an instance of PCP:

  $(*x_1, *y_1*)$, $(*x_1, y_1*)$, $(*x_2, y_2*)$, …, $(*x_k, y_k*)$, $(*\blacklozenge, \blacklozenge)$
- YES maps to YES?
  - given a match in original MPCP instance, can produce a match in the new PCP instance
- NO maps to NO?
  - given a match in the new PCP instance, can produce a match in the original MPCP instance

Post Correspondence Problem

- YES maps to YES?
  - given a match in original MPCP instance, can produce a match in the new PCP instance

x_1	x_4	x_5	x_2	x_1	x_3	x_4	x_4
y_1	y_4	y_5	y_2	y_1	y_3	y_4	y_4

$*x_1$	$*x_4$	$*x_5$	$*x_2$	$*x_1$	$*x_3$	$*x_4$	$*x_4$	$*\blacklozenge$
$*y_1*$	y_4*	y_5*	y_2*	y_1*	y_3*	y_4*	y_4*	$\blacklozenge$

Post Correspondence Problem

- NO maps to NO?
  - given a match in the new PCP instance, can produce a match in the original MPCP instance

can't match unless start with this tile

$*x_1$	$*x_4$	$*x_5$	$*x_2$	$*x_1$	$*x_3$	$*x_4$	$*x_4$	$*\blacklozenge$
$*y_1*$	y_4*	y_5*	y_2*	y_1*	y_3*	y_4*	y_4*	$\blacklozenge$

x_1	x_4	x_5	x_2	x_1	x_3	x_4	x_4
y_1	y_4	y_5	y_2	y_1	y_3	y_4	y_4

"*" symbols must align

can only appear at the end

Post Correspondence Problem

Theorem: PCP is undecidable.

Proof:

- show $A_{TM} \leq_m$ MPCP

  MPCP = $\{<(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)> :$
  $x_i, y_i \in \Sigma^*$ and there exists $(a_1, a_2, \ldots, a_n)$ for which $x_1 x_{a_1} x_{a_2} \ldots x_{a_n} = y_1 y_{a_1} y_{a_2} \ldots y_{a_n}\}$
- show MPCP $\leq_m$ PCP ☑

Post Correspondence Problem

Proof of $A_{TM} \leq_m$ MPCP:

- given instance of A_{TM}: <M, w>
- idea: a match will record an accepting computation history for M on input w
- start tile records starting configuration:
  - add tile (#, $\#q_0w_1w_2w_3\ldots w_n\#$)

$\left[\frac{\#}{\#q_0w_1w_2\ldots w_n\#}\right] = \left[\frac{\#C_1\#}{}\right]$

$\#$ / $\#C_1\#$

Post Correspondence Problem

$\left[\frac{\#}{\# q_0w_1w_2\ldots w_n\#}\right]\left[\frac{?}{?}\right]\left[\frac{?}{?}\right]\ldots\left[\frac{?}{?}\right] = \frac{\#C_1\#}{\#C_1\#C_2\#}$

- tiles for head motions to the right:
  - for all a,b $\in \Gamma$ and all q, r $\in$ Q with $q \neq q_{reject}$, if δ(q, a) = (r, b, R), add tile (qa, br)

$\left[\frac{qa}{br}\right]$

- tiles for head motions to the left:
  - for all a,b,c $\in \Gamma$ and all q, r $\in$ Q with $q \neq q_{reject}$, if δ(q, a) = (r, b, L), add tile (cqa, rcb)

$\left[\frac{cqa}{rcb}\right]$

Post Correspondence Problem

$\left[\frac{\#}{\# q_0w_1w_2\ldots w_n\#}\right]\left[\frac{?}{?}\right]\left[\frac{?}{?}\right]\ldots\left[\frac{?}{?}\right] = \frac{\#C_1\#}{\#C_1\#C_2\#}$

- tiles for copying (not near head)
  - for all a $\in \Gamma$, add tile (a, a)

$\left[\frac{a}{a}\right]$

- tiles for copying # marker
  - add tile (#, #)

$\left[\frac{\#}{\#}\right]$

- tiles for copying # marker and adding _ to end of tape
  - add tile (#, _#)

$\left[\frac{\#}{\_\#}\right]$

Post Correspondence Problem

$\left[\frac{\#}{\#uaq_{accept}v\#}\right]\left[\frac{?}{?}\right]\ldots\left[\frac{?}{?}\right] = \frac{\#uaq_{accept}v\#}{\#uaq_{accept}v\#uq_{accept}v\#}$

- tiles for deleting symbols to left of q_{accept}
  - for all a $\in \Gamma$, add tile (aq_{accept}, q_{accept})

$\left[\frac{aq_{accept}}{q_{accept}}\right]$

Post Correspondence Problem

$\left[\frac{\#}{\#q_{accept}av\#}\right]\left[\frac{?}{?}\right]\ldots\left[\frac{?}{?}\right] = \frac{\#q_{accept}av\#}{\#q_{accept}av\#q_{accept}v\#}$

- tiles for deleting symbols to right of q_{accept}
  - for all a $\in \Gamma$, add tile ($q_{accept}a$, q_{accept})

$\left[\frac{q_{accept}a}{q_{accept}}\right]$

Post Correspondence Problem

$\left[\frac{\#}{\#q_{accept}\#}\right]\left[\frac{?}{?}\right]\ldots\left[\frac{?}{?}\right] = \frac{\#q_{accept}\#\#}{\#q_{accept}\#\#}$

- tiles for completing the match
  - for all a $\in \Gamma$, add tile ($q_{accept}\#\#$, #)

$\left[\frac{q_{accept}\#\#}{\#}\right]$

Post Correspondence Problem

– YES maps to YES?
  - by construction, if M accepts w, there is a way to assemble the tiles to achieve this match:

$\#C_1\#C_2\#C_3\#\ldots\#C_m\#$
$\#C_1\#C_2\#C_3\#\ldots\#C_m\#$

where $\#C_1\#C_2\#C_3\#\ldots\#C_m\#$ is an accepting computation history

– NO maps to NO?
  - sketch: at any step if the "intended" next tile is not used, then it is impossible to recover and produce a match in the end (case analysis)

Post Correspondence Problem

We have proved:

Theorem: PCP is undecidable.

by showing:

– $A_{TM} \leq_m$ MPCP

– MPCP $\leq_m$ PCP

– conclude $A_{TM} \leq_m$ PCP

Beyond RE and co-RE

- We saw (by a counting argument) that there is *som*… that is neither RE or co-RE.

Therefore, not in co-RE

Therefore, not in RE

- We will prove this for a natural language:

  $EQ_{TM} = \{<M_1, M_2> : L(M_1) = L(M_2)\}$

- Recall:
  – A_{TM} is undecidable, but RE
  – co-A_{TM} is undecidable, but coRE