Today:
More Undecidability
The Computation History Method

10/14/05 1

Q] $001_{TM} = \{\langle M\rangle \mid M$ is a TM and $001 \in L(M)\}$

Show 001_{TM} is undecidable.

Proof] We show $A_{TM} \leq_m 001_{TM}$.

Give computable function f, st $f(\langle M,w\rangle) = \langle M'\rangle$ where $M' \in 001_{TM}$ iff $\langle M,w\rangle \in A_{TM}$.

M' = "On input 001
1) Simulate M on w,
2) If M halts, accept or reject according to M"

Reminder: we never actually need to run M', and we don't care what it does on other inputs. We just show that M' is constructible so that if we had a decider for 001_{TM}, we could run it on $\langle M'\rangle$ to decide A_{TM}

Q] What about $\overline{001_{TM}}$?

A] This is not decidable either, since decidability is closed under complement.

Remark: sometimes, it's easier to show that $\overline{L}$ is undecidable. Note: $A_{TM} \not\leq \overline{001_{TM}}$. why?

Q] $REG_{TM} = \{\langle M\rangle \mid M$ is a TM and $L(M)$ is regular$\}$

Show REG_{TM} is undecidable

Proof] Show $A_{TM} \leq REG_{TM}$.

We give computable f, st $f(\langle M,w\rangle) = \langle M'\rangle$ where $L(M')$ is regular iff M accepts w

M' = "on input x
1) If x is of the form 0^n1^n, accept
2) Simulate M on w, and if M halts, accept or reject according to M"

when M accepts w $\Longrightarrow$ $L(M') = \Sigma^*$, which is regular

when M rejects w $\Longrightarrow$ $L(M') = \{0^n 1^n \mid n \geq 0\}$, which is not regular. □

Rice's Theorem:

Any language of the form $B = \{\langle M \rangle \mid M$ is a TM and $L(M)$ has some property$\}$ is undecidable, Provided

i) The property applies to some, not all $\langle M \rangle$

ii) The property is only about the language of M.

i.e., if $L(M) = L(N)$, then either both $\langle M \rangle$ and $\langle N \rangle$ are in A, or both are not in A.

Proof| Assume that the empty language does not have the property (otherwise, work with $\bar{A}$)

Let T be some machine where $L(T)$ does have the property

Reduce $A_{TM} \leq_M B$ as follows

$f(\langle M, w \rangle) = M'$, where

M' = "on input x,

1) Simulate M on w. If it halts and rejects, reject.

2) If M accepts, run T on x"

when M accepts w $\Longrightarrow$ $L(M') = L(T)$, so M' has property

when M rejects w $\Longrightarrow$ $L(M') = \phi$, so M' doesn't have property

---

Computation History Method

Q| Let $ALL_{PDA} = \{\langle P \rangle \mid P$ is a PDA and $L(P) = \Sigma^*\}$

Show ALL_{PDA} is undecidable.

(Note: looks similar to E_{PDA}, but E_{PDA} is decidable)

Proof| Reduce $\overline{A_{TM}} \leq ALL_{PDA}$

Give $f(\langle M, w \rangle) = \langle P \rangle$

P accepts all invalid or non-accepting computation histories for M on w

How does P work? P looks for mistakes in the comp history

$$\#\underbrace{\qquad\qquad}_{C_1}\#\underbrace{\qquad\qquad}_{C_2^R}\#\cdots\#\underbrace{\qquad\qquad}_{C_K}\#$$

P must use non-determinism

1) One branch checks if C_1 is not a valid initial condition for M on w. If it is <u>not</u>, P accepts
2) another branch checks if C_K is not a valid end condition. Accepts if it is <u>not</u> (e.g., if it contains no accept state q_a)
3) other branches check if C_{i+1} incorrectly follows from C_i

a) Nondeterministically guess a C_i. Push it on to stack

b) If we assume that every other C_i is written backwards, then we pop from the stack as we read C_{i+1}. We compare it with C_i and accept if we find it inconsistent with M's transition function. □

Alternate solution, without assuming every other C_i is written backwards:

To check for an inconsistency between C_i and C_{i+1},

1) read from C_i while pushing characters on the stack, until we
2) <u>Guess</u> where in C_i the inconsistency will occur
3) Read from C_{i+1} while popping from the stack, until the stack is empty
4) If the current spot in C_{i+1} is indeed inconsistent from where we left off in C_i, <u>accept</u>