Lecture 28

Homework #28: 4.5.1 a, 4.5.1 b, 4.5.2, 4.5.3

Nondeterministic Turing Machines: provide a choice of actions for certain combinations of state and symbol.

Def. A **nondeterministic TM** is a quintuple $(K, \Sigma, \Delta, s, H)$, where

K, Σ, s, and H are defined as they are for the standard TM and

Δ is a finite subset of

$$((K-H) \times \Sigma) \times (K \times (\Sigma-\{>\} \cup \{\leftarrow,\rightarrow\}))$$

For the moment, let's view **NTMs only as acceptors**, not as deciders or computers.

We'll see that they add nothing over standard deterministic TMs.

Def. A string w is **accepted** by a NTM M if some computation of M on w halts.

Ex. L = {w : w has a substring with > 1 occurrence in w}

step 1. guess the substring
step 2. scan w for a copy of the substring
step 3. halt only if you find a match

Ex. L = all numbers that have an integer positive square root

guess the int pos sq rt; square it and check against input

Ex. L = all 2nd degree polynomials that have roots > 0

i.e., polynomials $ax^2 + bx + c$ such that
$ax^2 + bx + c = 0$ has positive valued solutions.

input:

#	a	$	b	$	c	#

$ is used rather than # for reasons that will be clear soon.

step 1. guess 2 solutions
step 2. compute the function on both guessed solns
step 3. compare the 2 computations to 0

halt iff the 2 solutions are 0.

Lemma. For every NTM M_1, we can construct a standard TM M_2 such that for any string w not containing #

(a) if M_1 halts on input w, then M_2 halts on input w.
(b) if M_1 doesn't halt on input w, then M_2 doesn't halt on input w.

Proof.

We'll first construct a 3-tape deterministic TM (which we know can be simulated by a standard TM!)

We know that for any state and tape symbol of M_1, there is a finite number of choices for "next move." These can be numbered 1, 2, . . . Let r be the maximum number of choices for any tape-symbol pair. Then any finite sequence of choices can be represented by a sequence of the digits 1-r. (Note that not all such sequences will represent valid choices of moves, since there may be fewer than r choices in some situations.)

M_2 will have 3 tapes:

(1) will hold the input

(2) on (2), M_2 will generate sequences of digits 1-r in a systematic manner

(3) on (3), M_2 will simulate M_1 on the input, using the sequence of steps on tape (2)

if M_1 enters a halt state, then M_2 will eventually halt as well;
if no sequence of moves leads to a halt state in M_1, then M_2 will just continue to generate move sequences infinitely.

<u>Theorem.</u> Any language accepted by a NTM is accepted by a deterministic TM.

Now, what does all of this mean with respect to deciding and computing?

<u>Def.</u> Let $M = (K, \Sigma, \Delta, s, \{y, n\})$ be a nondeterministic TM. We say that M decides a language $L \subseteq (\Sigma - \{>, \#\})^*$ if the following two conditions hold for all $w \in (\Sigma - \{>, \#\})^*$:

a. There is a natural number N, depending on M and w, such that there is no configuration C satisfying $(s, >\#w) \vdash^N C$.
b. $w \in L$ iff $(s, >\#w) \vdash^* (y, u\underline{a}v)$ for some $u, v \in \Sigma^*$, $a \in \Sigma$.

The first of these conditions specifies that the TM always halts.
The second says that we say yes as long as one of the computations says yes.

We say that a NDTM $M = (K, \Sigma, \Delta, s, \{h\})$ computes a function $f: (\Sigma - \{>, \#\})^* \to (\Sigma - \{>, \#\})^*$ if the following two conditions hold for all $w \in (\Sigma - \{>, \#\})^*$:

a. There is an N, depending on M and w, such that there is no configuration C satisfying $(s, >\#w) \vdash^N C$.
b. $(s, >\#w) \vdash^* (h, u\underline{a}v)$ iff $ua = >\#$, and $v = f(w)$.