Nondeterministic TMs

Goal: Define nondet-TMs and show that they are equivalent in power to standard-TMs.

A non-deterministic TM is the std-TM except that the transition function δ is now a transition relation; i.e. we can have

$$(q, \ell) \longrightarrow (q_1, \ell_1, L \text{ or } R)$$

<u>and</u>

$$(q, \ell) \longrightarrow (q_2, \ell_2, L \text{ or } R)$$

where $q_1 \neq q_2$ or $\ell_1 \neq \ell_2$ or one is L and other is R.

(still only finitely many possibilities)

We can't talk about the TM computing some output (b/c - which choice does it take?), but we can talk about the ~~TM~~ nondet-TM accepting or rejecting an input.

Same as before - we are biased to accepting - if <u>any</u> choice leads to acceptance, we accept the string. If <u>every</u> choice leads to rejection, we reject.

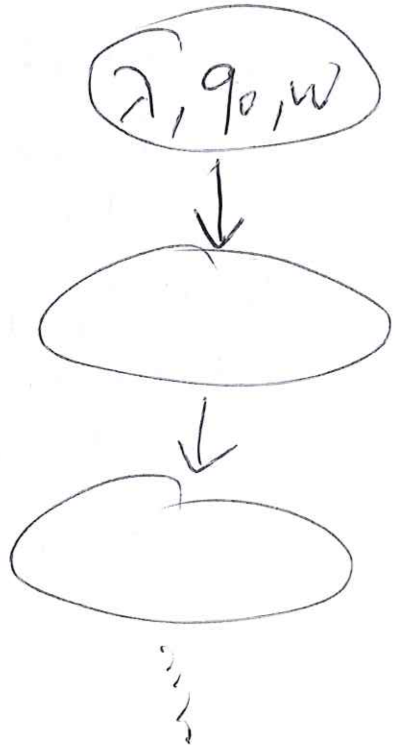
More formally, we still have the same notion of $(w, q, v) \longmapsto (w', q', v')$ (an instant descr yielding another in one step), and we accept a string w if there exists a seq. $(\lambda, q_0, w) \vdash (\quad) \vdash \cdots \vdash (v, q_f, u)$, where v, u are in Γ^*, $q_f \in F$.

~~For TM~~

Thm: Non-det TMs are no more powerful than standard TMs.

Pf: I need to give an method that, given a non-det TM M, constructs a standard TM $\hat{M}$ that accepts exactly the same strings.

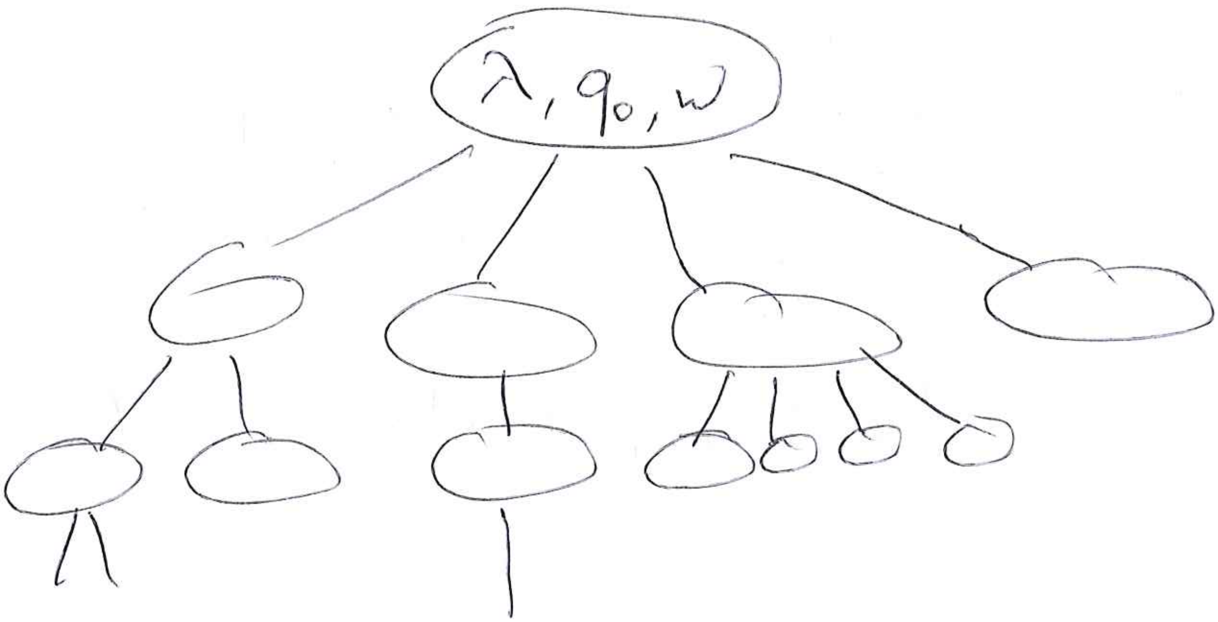
Picture - a computation on a standard TM is a single path of instant descr.



This path can be infinite (which means our TM never halts).

If the path ends, and ends in a final state, we accept.

A ~~no~~ computation on a non-det TM is a (possibly infinitely tall, but <u>not</u> infinitely wide tree).



If there is a ~~leaf (node w/ no children)~~ node in ~~a~~ the tree which has a final state, then we accept.

So - to find out if we have an ~~accept~~ a final state somewhere, we can do a breadth-first search on this tree (Not a depth-first search b/c some branches might be infinitely long, ~~&~~ and we don't want to just keep going down that branch)

How do we program this on a TM?

(i.e. how to make a standard-TM do a breadth-first search on this tree?)

We'll make a 3-head-3-tape TM (called $\tilde{M}$) - we "know" how to make a standard-TM $\hat{M}$ from $\tilde{M}$.

Tape 1: The original input ~~on~~ it never changes.

Tape 2: The computation going down whichever path ~~we~~ in the tree we're doing currently.

Tape 3: Specification of our current tree branch:

(e.g. choice 3, then choice 6, then choice 27, then choice 4 (say, out of 50 possible)

We do the following loop:

① Increment tape 3 (in the order (—), (choice 1), (choice 2), ..., (choice 50), (choice 1, then 1), (choice 1, then 2), (1,3), ..., (1,50), (2,1), ..., (2,50), ----, (50,50), (1,1,1), ...)

---

② Copy tape 1 to tape 2, and run the non-det-TM on tape 2 w/ the specified choices read from tape 3, (and move head on tape 3 to ~~do~~ next choice symbol) — until you have run out of choice specifications on tape 3. (or you have halted b/c you're trying to do a nonexistent choice)

③ If you are in a final state, celebrate - otherwise go back to ①.

This 3-head 3-tape TM eventually accepts the string if some comput. of the non-det TM accepts. If the non-det TM doesn't accept, it keeps running forever.

So our TM $\tilde{M}$ ~~a~~ accepts the same strings as M.