Variants of TM's

Many different ways to define TM model:
- multiple tapes, 2-way tapes, nondeterminism

Our model, and all reasonable variants, have same power (i.e., accept same class of languages)

So the model is robust

Ex. \( \delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L,R,S\} \) \hspace{1cm} (S for stay)

Is this class of TM more powerful? NO!

Given any TM M' in extended class, can define
a regular TM M that does 2 moves (R,L)
for every S move of M', and acts exactly like M' otherwise

Two-way tapes

Lemma: If L is accepted by a TM M with a 2-way tape, then L is accepted by a TM M' with 1-way tape.

Proof idea: Use 2 tracks:

\[
\begin{array}{c|c}
A_0 & A_1 \\
\hline
$ & A_1 \\
A_0 & $ \\
\end{array}
\]

for single 2-way:

\[
\begin{array}{c|c|c}
A_1 & A_0 & A_1 \\
\hline
\ldots
\end{array}
\]

M' must keep track of whether scanning symbol on top or bottom track, build into state:

\( Q' = \{q_1\} \cup \{Q \times \{\bot\}\} \) \hspace{1cm} \( \Gamma' = \{[x,y] | x, y \in \Gamma \text{ or } y = $\} \)

\( \Sigma' = \{[a\square] | a \in \Sigma \} \)
Multiple tapes

\[ \delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L,R\}^k \]

Lemma: If L is accepted by a multitape TM, then L is accepted by a single tape TM.
Proof idea: use tracks, with special markers for each tape head.

<table>
<thead>
<tr>
<th>w</th>
<th>x</th>
<th>y</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>b</td>
<td>m n</td>
</tr>
</tbody>
</table>

M' will simulate M by making multiple passes over tape, 1 for each track. First, it needs to determine all the symbols being read, in order to determine the next move. So \( k \) symbols must be stored in state. Then, once the next move is determined, it must update each track with new symbol and new tape head position.

Nondeterministic TM's

\[ \delta : Q \times \Gamma \rightarrow P( Q \times \Gamma \times \{L,R\} ) \]

Lemma: If L is accepted by a nondet. TM N, then L is accepted by some deterministic TM D.
Proof idea: D will simulate N by trying all possible branches of N's computation on input w. If D finds an accept state on some branch, it accepts. Otherwise, D does not accept.

QU: How should D search the computation tree??

How represent a branch?
<table>
<thead>
<tr>
<th>Robustness of TM's</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Is this surprising??</strong></td>
</tr>
<tr>
<td>TM's ←→ &quot;Idealized&quot; programming languages</td>
</tr>
<tr>
<td>Programming languages, if sufficiently general, can do the same computations</td>
</tr>
<tr>
<td>Ex. C, Lisp, Java</td>
</tr>
<tr>
<td><strong>TM's make formal our informal notion of algorithm</strong></td>
</tr>
<tr>
<td>Church-Turing Thesis: The Turing machine model</td>
</tr>
<tr>
<td>(and any reasonable variant of it) embodies our informal notion of algorithm.</td>
</tr>
<tr>
<td>Can’t be proved!</td>
</tr>
<tr>
<td><strong>From now on, TM ↔ Algorithms</strong></td>
</tr>
</tbody>
</table>