Section: Other Models of Turing Machines

Definition: Two automata are equivalent if they accept the same language.

Turing Machines with Stay Option

Modify $\delta$,

Theorem Class of standard TM’s is equivalent to class of TM’s with stay option.

Proof:

$\bullet (\Rightarrow)$: Given a standard TM $M$, then there exists a TM $M'$ with stay option such that $L(M) = L(M')$. 
(\iff):

Given a TM M with stay option, construct a standard TM M' such that L(M) = L(M').

\[ M = (Q, \Sigma, \Gamma, \delta, q_0, B, F) \]

\[ M' = \]

For each transition in M with a move (L or R) put the transition in M'. So, for

\[ \delta(q_i, a) = (q_j, b, L \text{ or } R) \]

put into \( \delta' \)

For each transition in M with S (stay-option), move right and move left. So for

\[ \delta(q_i, a) = (q_j, b, S) \]

\[ L(M) = L(M'). \text{ QED.} \]
Definition: A multiple track TM divides each cell of the tape into k cells, for some constant k.

A 3-track TM:

<table>
<thead>
<tr>
<th></th>
<th>b</th>
<th>c</th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>a</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A multiple track TM starts with the input on the first track, all other tracks are blank.

δ:
Theorem Class of standard TM’s is equivalent to class of TM’s with multiple tracks.

Proof: (sketch)

• \((\Rightarrow)\): Given standard TM \(M\) there exists a TM \(M'\) with multiple tracks such that \(L(M)=L(M')\).

• \((\Leftarrow)\): Given a TM \(M\) with multiple tracks there exists a standard TM \(M'\) such that \(L(M)=L(M')\).
Definition: A TM with a semi-infinite tape is a standard TM with a left boundary.

Theorem Class of standard TM’s is equivalent to class of TM’s with semi-infinite tapes.

Proof: (sketch)

\( \Rightarrow \): Given standard TM M there exists a TM M’ with semi-infinite tape such that \( L(M) = L(M') \).

Given M, construct a 2-track semi-infinite TM M’
\( \text{TM } M \)

\[
\begin{array}{ccc}
\cdots & a & b & c & \cdots \\
\end{array}
\]

\( \text{TM } M' \)

\[
\begin{array}{cccc}
# & a & b & c \\
# \\
\end{array}
\]

\( \cdots \leftarrow \text{right half} \)

\( \leftarrow \text{left half} \)

\( (\iff) : \text{Given a TM } M \text{ with semi-infinite tape there exists a standard TM } M' \text{ such that } L(M) = L(M') \).
Definition: An Multitape Turing Machine is a standard TM with multiple (a finite number) read/write tapes.

For an $n$-tape TM, define $\delta$: 
Theorem Class of Multitape TM’s is equivalent to class of standard TM’s.

Proof: (sketch)

• (⇐): Given standard TM M, construct a multitape TM M’ such that $L(M) = L(M')$.

• (⇒): Given n-tape TM M construct a standard TM M’ such that $L(M) = L(M')$.

<table>
<thead>
<tr>
<th></th>
<th>#</th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>a</td>
<td>a</td>
<td>a</td>
<td>a</td>
</tr>
<tr>
<td>#</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>b</td>
<td>b</td>
<td>b</td>
<td>b</td>
</tr>
<tr>
<td>#</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Definition: An Off-Line Turing Machine is a standard TM with 2 tapes: a read-only input tape and a read/write output tape.

Define $\delta$:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

input tape (read only)

Control Unit

<table>
<thead>
<tr>
<th></th>
<th>b</th>
<th>b</th>
<th>d</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

read/write tape
Theorem Class of standard TM’s is equivalent to class of Off-line TM’s.

Proof: (sketch)

• \((\Rightarrow)\): Given standard TM \(M\) there exists an off-line TM \(M’\) such that \(L(M) = L(M’).\)

• \((\Leftarrow)\): Given an off-line TM \(M\) there exists a standard TM \(M’\) such that \(L(M) = L(M’).\)
Running Time of Turing Machines

Example:

Given $L = \{a^n b^n c^n | n > 0\}$. Given $w \in \Sigma^*$, is $w$ in $L$?

Write a 3-tape TM for this problem.
Definition: An Multidimensional-tape Turing Machine is a standard TM with a multidimensional tape

Define $\delta$: 
Theorem Class of standard TM’s is equivalent to class of 2-dimensional-tape TM’s.

Proof: (sketch)

• ($\Rightarrow$): Given standard TM $M$, construct a 2-dim-tape TM $M'$ such that $L(M) = L(M')$.

• ($\Leftarrow$): Given 2-dim tape TM $M$, construct a standard TM $M'$ such that $L(M) = L(M')$. 
Construct $M'$

<table>
<thead>
<tr>
<th></th>
<th>#</th>
<th>a</th>
<th></th>
<th>#</th>
<th>b</th>
<th></th>
<th>#</th>
<th>c</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>1</td>
<td>1</td>
<td>#</td>
<td>1</td>
<td>1</td>
<td>#</td>
<td>1</td>
<td>1</td>
<td>#</td>
</tr>
</tbody>
</table>
Definition: A *nondeterministic* Turing machine is a standard TM in which
the range of the transition function is a set of possible transitions.

Define $\delta$:

Theorem Class of deterministic TM’s is equivalent to class of
nondeterministic TM’s.

Proof: (sketch)

- ($\Rightarrow$): Given deterministic TM $M$, construct a nondeterministic TM
  $M’$ such that $L(M) = L(M’)$. 

- ($\Leftarrow$): Given nondeterministic TM $M$, construct a deterministic TM
  $M’$ such that $L(M) = L(M’)$. Construct $M’$ to be a 2-dim tape
  TM.
A step consists of making one move for each of the current machines. For example: Consider the following transition:

$$\delta(q_0, a) = \{(q_1, b, R), (q_2, a, L), (q_1, c, R)\}$$

Being in state \(q_0\) with input abc.

<table>
<thead>
<tr>
<th></th>
<th>#</th>
<th>#</th>
<th>#</th>
<th>#</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>q0</td>
<td></td>
<td></td>
<td>#</td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td></td>
</tr>
</tbody>
</table>


The one move has three choices, so 2 additional machines are started.

<p>| | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
</tr>
<tr>
<td>#</td>
<td>b</td>
<td>b</td>
<td>c</td>
<td>#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td></td>
<td>q1</td>
<td></td>
<td>#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>a</td>
<td>b</td>
<td>c</td>
<td>#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>q2</td>
<td></td>
<td>#</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>c</td>
<td>b</td>
<td>c</td>
<td>#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td></td>
<td>q1</td>
<td>#</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
<td>#</td>
</tr>
</tbody>
</table>
Definition: A 2-stack NPDA is an NPDA with 2 stacks.

Define $\delta$: 
Consider the following languages which could not be accepted by an NPDA.

1. \( L = \{ a^n b^n c^n | n > 0 \} \)

2. \( L = \{ a^n b^n a^n b^n | n > 0 \} \)

3. \( L = \{ w \in \Sigma^* | \text{number of } a \text{'s equals number of } b \text{'s equals number of } c \text{'s} \}, \Sigma = \{ a, b, c \} \)
Theorem Class of 2-stack NPDA’s is equivalent to class of standard TM’s.

Proof: (sketch)

• \(\Rightarrow\): Given 2-stack NPDA, construct a 3-tape TM \(M'\) such that \(L(M) = L(M')\).
• ($\iff$): Given standard TM $M$, construct a 2-stack NPDA $M'$ such that $L(M) = L(M')$. 
Universal TM - a programmable TM

- **Input:**
  - an encoded TM $M$
  - input string $w$

- **Output:**
  - Simulate $M$ on $w$
An encoding of a TM

Let TM \( M = \{Q, \Sigma, \Gamma, \delta, q_1, B, F\} \)

- \( Q = \{q_1, q_2, \ldots, q_n\} \)
  - Designate \( q_1 \) as the start state.
  - Designate \( q_2 \) as the only final state.
  - \( q_n \) will be encoded as \( n \) 1’s

- Moves
  - \( L \) will be encoded by 1
  - \( R \) will be encoded by 11

- \( \Gamma = \{a_1, a_2, \ldots, a_m\} \)
  - where \( a_1 \) will always represent the \( B \).
For example, consider the simple TM:

\[ a; a, R \]

\[
\begin{array}{c}
q_1 \quad b; a, L \quad q_2
\end{array}
\]

\[ \Gamma = \{ B, a, b \} \] which would be encoded as

The TM has 2 transitions,

\[ \delta(q_1, a) = (q_1, a, R), \quad \delta(q_1, b) = (q_2, a, L) \]

which can be represented as 5-tuples:

\[ (q_1, a, q_1, a, R), (q_1, b, q_2, a, L) \]

Thus, the encoding of the TM is:

\[ 010110101101101011101101101010 \]
For example, the encoding of the TM above with input string “aba” would be encoded as:

01011010110110110110110100110110110110

Question: Given \( w \in \{0, 1\}^+ \), is \( w \) the encoding of a TM?
Universal TM

The Universal TM (denoted $M_U$) is a 3-tape TM:

```
1111
```

tape contents of $M$

```
0110...
```

encoding of $M$

```
0101...
```

current state of $M$

```
Control
Unit
```

Diagram: A control unit with outgoing arrows to tape contents of $M$, encoding of $M$, and current state of $M$. Each state is represented by a sequence of digits.
Program for $M_U$

1. Start with all input (encoding of TM and string $w$) on tape 1. Verify that it contains the encoding of a TM.

2. Move input $w$ to tape 2

3. Initialize tape 3 to 1 (the initial state)

4. Repeat (simulate TM M)
   (a) consult tape 2 and 3, (suppose current symbol on tape 2 is $a$ and state on tape 3 is $p$)
   (b) lookup the move (transition) on tape 1, (suppose $\delta(p,a) = (q,b,R)$.)
   (c) apply the move
      • write on tape 2 (write $b$)
      • move on tape 2 (move right)
      • write new state on tape 3 (write $q$)
Observation: Every TM can be encoded as string of 0’s and 1’s.

Enumeration procedure - process to list all elements of a set in ordered fashion.

Definition: An infinite set is countable if its elements have 1-1 correspondence with the positive integers.

Examples:

- $S = \{ \text{positive odd integers} \}$
- $S = \{ \text{real numbers} \}$
- $S = \{ w \in \Sigma^+ \}, \Sigma = \{a, b\}$
- $S = \{ \text{TM’s} \}$
- $S = \{ (i,j) \mid i,j>0, \text{are integers} \}$
Linear Bounded Automata

We place restrictions on the amount of tape we can use,

\[
\begin{array}{ccc}
  & a & b & c \\
\hline
\uparrow & & & \\
\end{array}
\]

Definition: A linear bounded automaton (LBA) is a nondeterministic TM
\(M=(Q,\Sigma, \Gamma, \delta, q_0, B, F)\) such that \([,]\) \(\in\Sigma\) and the tape head cannot move out of the confines of []’s. Thus,
\(\delta(q_i, [) = (q_j, [, R)\), and \(\delta(q_i, ]) = (q_j, ], L)\)

Definition: Let \(M\) be a LBA.
\(L(M)=\{w \in (\Sigma - \{[, ]\})^* | q_0[w] \vdash [x_1q_fx_2]\}\)

Example: \(L=\{a^nb^nc^n | n > 0\}\) is accepted by some LBA