Turing Machines

1. Infinite tape, divided into cells.
2. Read/Write Head
3. Finite Number of States
4. In each step, head can read/write and move left/right.

Example:

Suppose we want to check if the input contains same number of a's as b's.

State	a	b	B	X
q0	q1, X, R	q2, X, R	qA,B,R	q0, X, R
q1	q1, a, R	q3, X, L		q1, X, R
q2	q3, X, L	q2, b, R		q2, X, R
q3	q3, a, L	q3, b, L	q0,B,R	q3, X, L
qA				

Turing Machines

1. Function Computation
2. Language Acceptance

Turing Machines

Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$.

Q: a set of states

Σ: input alphabet set

Γ: tape alphabet. $\Sigma \subseteq \Gamma$.

δ: transition function from $Q \times \Gamma$ to $Q \times \Gamma \times \{L, R\}$.

q_0: starting state

B: blank symbol. We assume $B \in \Gamma - \Sigma$

F: set of final/accepting states. $F \subseteq Q$.

Usually, input is given without any blanks in between.

Instantaneous Description

1. We leave out blanks on both ends.
Exception: if head is among the blanks .....

2. $x_0 x_1 \ldots x_{n-1} q x_n x_{n+1} \ldots x_m$.

3. $x_0 x_1 \ldots x_{n-1} q x_n x_{n+1} \ldots x_m \vdash$ next ID

4. $\vdash^*$ can be defined by saying 'zero or more steps'.
$ID_0 \vdash ID_1 \vdash \ldots \vdash ID_n$, then
$ID_0 \vdash^* ID_n$.
(Here n maybe 0).

Language Accepted by Turing Machine

TM accepts x , if

$$q_0 x \vdash^* \alpha q_f \beta$$

where $q_f \in F$.
$L(M) = \{x : q_0 x \vdash^* \alpha q_f \beta$, for some $q_f \in F\}$.

Languages/Functions

1. A language L is said to be *recursively enumerable* (RE), (computably enumerable, CE) if some Turing Machine accepts the language L.

2. A language L is said to be *recursive* (*decidable*), if some Turing Machine accepts the language L, and Halts on all the inputs.

3. A function f is said to be *partial recursive* (partial computable), if some Turing Machine computes the function (it halts on all the inputs on which f is defined, and it does not halt on inputs on which f is not defined).

4. A function f is said to be *recursive* (computable), if some Turing Machine computes the function, and f is defined on all elements of Σ^*.

Turing Machine and Halting

Machine may never halt.
Cannot determine if a machine will halt on a particular input ....

Modifications of Turing Machines

– Stay where you are
– memorize a constant amount of information
– Multi Track Turing Machines
– Semi-Infinite Tapes
– Multi Tape Turing Machines
– Nondeterministic Turing Machines

Simulation of multi-track (one way infinite tape) TM

Simulation for 2 tracks.
Generalization to multitrack is easy.
Suppose $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$, a 2-track (one tape) TM is given.
Note:
$\delta(q, x, y) = (p, u, v, m)$, means if the machine is in state q, reading x on first track and y on second track, then
— next state is p, u is written on the first track, v is written on the second track, and m is the movement of head (L, R or S).
We construct $M' = (Q', \Sigma', \Gamma', \delta', q_0', B', F')$, a one track, one tape, TM as follows.

Σ': For each $x, y \in \Sigma$ we have $[x, y], [x, B], [B, y] \in \Sigma'$.
Γ': For each $x, y \in \Gamma$ we have $[x, y] \in \Gamma'$.
$B' = [B, B]$.
$Q' = Q$.
$F' = F$.
$q_0' = q_0$.
Input: Any input of the form $(x_1, y_1)(x_2, y_2) \ldots (x_n, y_n)$ for M is mapped to input $[x_1, y_1][x_2, y_2] \ldots [x_n, y_n]$ for M'.
Note: In the above input $(x_1, y_1)(x_2, y_2) \ldots (x_n, y_n)$ for M, means that $x_1, x_2, \ldots, x_n$ is on the first track, and $y_1, y_2, \ldots, y_n$ is on the second track.
δ' is defined as follows: for $m \in \{S, L, R\}$, if $\delta(q, x, y) = (p, u, v, m)$, then $\delta'(q, [x, y]) = (p, [u, v], m)$.
It is easy to verify that any instantaneous description of the form:
$(x_1, y_1), (x_2, y_2), ..., (x_l, y_l)q(x_{l+1}, y_{l+1}), \ldots$ is mapped to
$[x_1, y_1], [x_2, y_2], ..., [x_l, y_l]q[x_{l+1}, y_{l+1}], \ldots$.

Simulation of TM with two-way infinite tape using TM with one way infinite tape

Suppose $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$, a TM with 2-way infinite tape, is given.
We construct $M' = (Q', \Sigma', \Gamma', q'_0, B', F')$, a TM with one way infinite tape, as follows.
Σ': For each $x \in \Sigma$ we have $[x, B] \in \Sigma'$.
Γ': For each $x, y \in \Gamma$ we have $[x, y] \in \Gamma'$. In addition, for each $x \in \Gamma$, we have $[x, \$] \in \Gamma'$. Here \$ is a special symbol not in Γ.
$B' = [B, B]$.
Input: Each input $x_1, x_2, \ldots, x_n$ for M is mapped to input $[x_1, B], [x_2, B] \ldots,$ for M'.
Q': For each $q \in Q$, we have $[q, U]$ and $[q, D]$ in Q'. In addition we have a special state called q_{new} in Q'.
F': for each $q \in F$, we have $[q, U]$ and $[q, D]$ in F'.
$q'_0 = q_{new}$.

δ' is defined as follows:
$\delta'(q_{new}, [x, B]) = ([q_0, U], [x, \$], S)$. ($\delta'(q_{new}, [x, y])$ for $y \neq B$ is not defined; we will not be needing it).

Suppose $x \in \Gamma$. Consider any $q \in Q$. Then δ' for remaining states, symbols in $Q' \times \Gamma'$ is defined as follows.

1. Suppose $x, y, w \in \Gamma$.
   If $\delta(q, x) = (p, y, S)$, then
   $\delta'([q, U], [x, w]) = ([p, U], [y, w], S)$, and $\delta'([q, D], [w, x]) = ([p, D], [w, y], S)$.
   If $\delta(q, x) = (p, y, R)$, then
   $\delta'([q, U], [x, w]) = ([p, U], [y, w], R)$, and $\delta'([q, D], [w, x]) = ([p, D], [w, y], L)$.
   If $\delta(q, x) = (p, y, L)$, then
   $\delta'([q, U], [x, w]) = ([p, U], [y, w], L)$, and $\delta'([q, D], [w, x]) = ([p, D], [w, y], R)$.

2. Suppose $x, y \in \Gamma$.

$$\text{If } \delta(q, x) = (p, y, S), \text{ then } \delta'([q, U], [x, \$]) = ([p, U], [y, \$], S).$$
$$\text{If } \delta(q, x) = (p, y, R), \text{ then } \delta'([q, U], [x, \$]) = ([p, U], [y, \$], R).$$
$$\text{If } \delta(q, x) = (p, y, L), \text{ then } \delta'([q, U], [x, \$]) = ([p, D], [y, \$], R).$$

3. Suppose $x \in \Gamma$.

$$\delta'([q, D], [x, \$]) = ([q, U], [x, \$], S).$$

Exercise: What is the correspondence between ID of M and ID of M'?

Exercise: Give details of how to simulate a multi-tape TM using one tape TM.

Church-Turing Thesis

Whatever can be computed by an algorithmic device (in function computation sense, or language acceptance sense) can be done by a Turing Machine.

Codings of TMs/Strings; Gödel Numbering

States: $q_1, q_2, \ldots$ are the states, with q_1 being start state and q_2 the only accepting state.

Tape symbols: $X_1, X_2, \ldots, X_s$ are tape symbols. X_1 is 0, X_2 is 1 and X_3 is blank.

Directions: L is D_1 and R is D_2.

Coding Transition: $\delta(q_i, X_j) = (q_k, X_l, D_m)$, then code it using string $0^i 1 0^j 1 0^k 1 0^l 1 0^m$.
(Note that each of i, j, k, l, m is at least 1).

Code of TM is: $C_1 11 C_2 11 C_3 \ldots C_n$, where C_i are the codes of all the transitions in the TM.

For a string x over $\{0, 1\}^*$, let $1x$ (in binary) -1 be its code.
Similarly, for larger alphabets.

M_i denotes the Turing Machine with code number i.
$W_i = L(M_i)$ denotes the language accepted by Turing Machine with code number i.
φ_i denotes the function computed by the i-th Turing Machine.
Without loss of generality, we often take $W_i = L(M_i) = domain(M_i)$.

A non-RE language

Let $L_d = \{w_i : w_i \notin L(M_i)\}$.

Pairing Function

Bijection from $N \times N$ to N.
$\langle x, y \rangle = 2^x(2y+1) - 1$.
One can extend it to triples by using $\langle x, y, z \rangle = \langle x, \langle y, z \rangle \rangle$.
Extend to coding m-tuples N^m to N.

Universal Turing Machine

$L_u = \{\langle i, w\rangle : M_i \text{ accepts } w\}.$