Turing Machines

Finite control and an infinite tape.

Initially the input is placed on the tape, the head of the tape is reading the first symbol of the input, and the state is q_0.

The other squares contain blanks.

$(Q, \Sigma, \Gamma, \delta, q_0, B, F)$

$\Sigma \cup \{B\} \subseteq \Gamma$

$\delta(q, X) = (p, Y, D)$ where $D = \leftarrow$ or $\rightarrow$.

B is the official blank symbol.

Instantaneous Description (ID): $X_1 X_2 \ldots X_{i-1} q X_i X_{i+1} \ldots X_n$, meaning that the state is q, the head is scanning the i-th symbol from the left. To the left of X_1 and the right of X_n only B's.

Suppose $\delta(q, X_i) = (p, Y, \leftarrow)$, then

$$X_1 X_2 \ldots X_{i-1} q X_i X_{i+1} \ldots X_n \vdash X_1 X_2 \ldots X_{i-2} p X_{i-1} Y X_{i+1} \ldots X_n$$

$L(M) = \{w \in \Sigma^* | q_0 w \vdash^* \alpha p \beta, p \in F\}$

Languages accepted by TMs are called **recursively enumerable (RE)**.

We assume that a TM halts as soon as it enters an accepting state.

A nondeterministic TM may have a choice of moves:

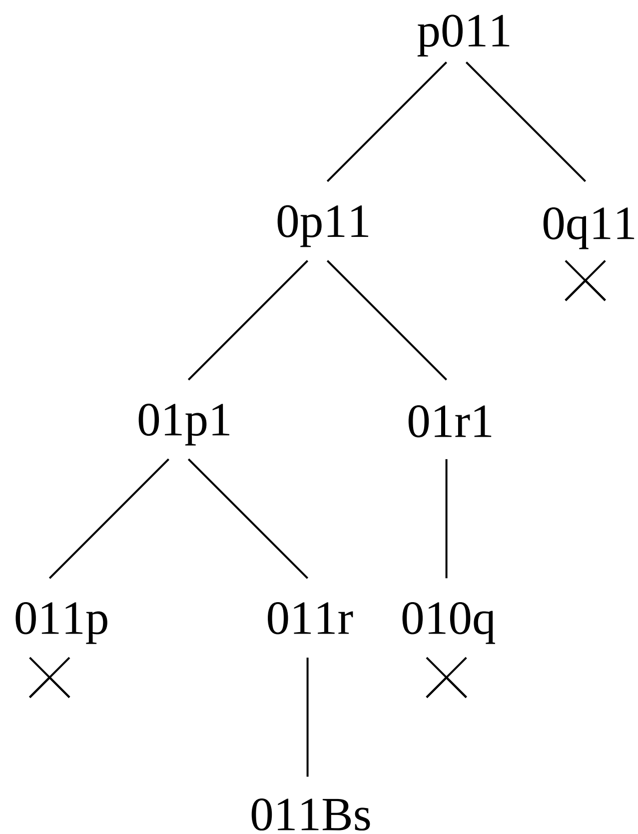
$$\delta(q, X) = \{(q_1, Y_1, D_1), (q_2, Y_2, D_2), \ldots, (q_k, Y_k, D_k)\}$$

Thm: If N is a nondet TM, then there exists a det TM D such that $L(N) = L(D)$.

Proof: D tries out all the possible moves of N, using "breadth-first" search.

Ex. TM N s.t. $L(N) = \{w \in \{0,1\}^* |$ last symbol of w is 1 $\}$.

$\delta(p,0) = \{(p,0,\rightarrow),(q,0,\rightarrow)\}$, $\delta(p,1) = \{(p,1,\rightarrow),(r,1,\rightarrow)\}$,
$\delta(r,B) = \{(s,B,\rightarrow)\}$, $\delta(r,0/1) = \{(q,0,\rightarrow)\}$, $F = \{s\}$, $q_0 = p$.



Different variants of TMs are equivalent (*robustness*): tape infinite in only one direction, or several tapes.

We can now give more details for the proof of thm on previous slide. D maintains a sequence of ID's on tape 1:

$\cdots$	ID_1	ID_2	ID_3^*	$\cdots$

and uses a second tape for scratch work.

The marked ID (*) is the current ID. D copies it to the second tape, and examines it to see if it is accepting. If it is, it accepts.

If it is not, and N has k possible moves, D copies the k new ID's resulting from these moves at the end of tape 1, and marks the next ID as current.

If max nr of choices of N is m, and N makes n moves, D examines $1 + m + m^2 + m^3 + \cdots + m^n \approx nm^n$ many IDs.

Undecidability

We can encode every Turing machine with a string over $\{0, 1\}$. For example, if M is a TM:

$$(\{q_1, q_2\}, \{0, 1\}, \delta, \ldots)$$

and $\delta(q_1, 1) = (q_2, 0, \rightarrow)$ is one of the transitions, then it could be encoded as:

$$\underbrace{0}_{q_1}\ 1\ \underbrace{00}_{1}\ 1\ \underbrace{00}_{q_2}\ 1\ \underbrace{0}_{0}\ 1\ \underbrace{0}_{\rightarrow}\ 11\ \underbrace{\ldots\ldots\ldots\ldots\ldots\ldots}_{\text{encoding of other transitions}}$$

Not every string is going to be a valid encoding of a TM (for example the string 1 does not encode anything in our convention).

Let all "bad strings" encode a default TM M_{default} which has one state, and halts immediately, so $L(M_{\text{default}}) = \emptyset$.

Let the following be a list of all strings of 0s and 1s (in *lexicographic* ordering):

$$0, 1, 00, 01, 10, 11, 000, 001, 010, 100, 101, \ldots$$

Let $w_i \in \{0,1\}^*$ be the i-th string on this list:

$$w_1, w_2, w_3, w_4, w_5, \ldots, w_i, \ldots$$

For example, $w_8 = 001$.

Then, let M_i be the Turing machine encoded by w_i. If w_i is a valid encoding, then M_i is the corresponding TM, but if w_i is garbage, then $M_i = M_{\text{default}}$.

In other words:

$$M_i = \begin{cases} \text{TM described by } w_i & \text{if } w_i \text{ is valid} \\ M_{\text{default}} & \text{otherwise} \end{cases}$$

For example, $M_2 = M_{\text{default}}$ as $w_2 = 1$ which is not a valid encoding of a TM.

Thus, we have the corresponding list of TMs:

$$M_1, M_2, M_3, M_4, M_5, \ldots, M_i, \ldots$$

Any TM under the sun is somewhere in this list.

A language L is **recursive** if there exists a TM M that halts in an accepting state for all $x \in L$, and halts in a non-accepting state for all $x \notin L$.

In other words, L is recursive if there exists a TM M such that $L = L(M)$, and M *halts on all inputs*.

$$Q \supseteq \text{ Halting } = \text{ Accepting } \cup \text{ Rejecting } \cup \text{ Other Halting}$$

Recursive languages correspond to languages that can be recognized *algorithmically*.

A language L is **recursively enumerable (RE)** if there exists a TM M that halts in an accepting state for all $x \in L$. In other words, L is RE if there exists an M such that $L = L(M)$ (but M does not necessarily halt on all inputs).

L_d, the **diagonal language**, is the set of strings w_i such that M_i does not accept w_i.

In words, L_d is the set of strings which encode TM which do not accept their own encoding when presented to them as input.

Formally,

$$L_d = \{w | w = w_i \text{ and } M_i \text{ does not accept } w_i\}$$

Thm (0): L_d is not RE.

Proof: Suppose that $L_d = L(M)$, for some M. Since all TM under the sun are encoded somewhere on the list $w_1, w_2, w_3, \ldots$, there exists a w_i that encodes M, i.e., $M = M_i$.

Is $w_i \in L_d$?

If $w_i \in L_d$, then $M (= M_i)$ accepts w_i, but then by def of L_d, $w_i \notin L_d$.

If $w_i \notin L_d$, then $M (= M_i)$ does not accept w_i, so $w_i \in L_d$.

Contradiction, so our assumption that L_d is RE is wrong.