Turing machines 2

Example (Revisited), $L = \{0^m, 1^m | m \in N, m \geq 1\}$

WANT to write down a transition mapping for the Turing machine that recognizes L. Algorithm translated into a transition diagram

Transition mapping

We have accounted for all pieces of ? in our algorithm. Therefore we have specified a turing machine, where A = {0, 1}, $\tilde{A}$ = {0, 1, _, x, y}, $S\prime = \{i, Saccept, Sreject, S1, S2, S3...S5\}$.

- i is the initial state,
- Saccept is the accept state,
- Sreject is the reject state

We just have to write down the transition mapping.
$t.Sx\tilde{A} \to Sx\tilde{A}x\{l, R\}$
$t(i, 0) = (S1, x, R)$
$t(i, 1) = (Sreject, 1, L)$
$t(i, \_) = (Sreject, \_, L)$
These above are the only three transitions possible out of state i, but $t.Sx\tilde{A} \to Sx\tilde{A}xL, R$. So technically to write down the full transition mapping we must assign triplets in $S\prime x\tilde{A}x\{L, R\}$ even to inputs from $\tilde{A}$ that cannot occur when in i. Technically, the turing machine halts when it enters either an accepting state (Saccept) or a rejecting state (Sreject), so in practice we can define $\tilde{S} = \{i, S1, S2....S5\} = S$
$\{Saccept, Sreject\}$, so we avoid writing down the transitions from Saccept and Sreject.

We have states $S1, S2...S5$ for which we still need to write down the transitions.
LEGEND R/L = direction of tape head movement. t = Transition.

- S1
  - t(s1, 0) = (S1, 0, R)
  - t(s1, 1) = (S2, y, L)
  - t(s1, y) = (S2, y, R)
  - t(s1, x) cannot occur
  - t(s1, _) cannot occur
- S2
  - t(S2, y) = (S2, y, R)
  - t(S2, 1) = (S3, y, L)
  - t(S2, 0) = (Sreject, 0, R)
  - t(S2, _) = (Sreject, _ R)
  - t(S2, x) cannot occur
- S3
  - t(S3, y) = (S3, y, L)
  - t(S3, 0) = (S3, 0, L)
  - t(S3, x) = (S4, x, R)
  - t(S3, _) cannot occur
  - t(S3, 1) cannot occur
- S4
  - t(S4, x) = (S4, X, R)
  - t(S4, y) = (S5, y, R)
  - t(S4, 0) = (S1, x, R)
  - t(S4, 1) cannot happen
  - t(S4, _) cannot happen
- S5
  - t(S5, y) = (S5, y, R)
  - t(S5, _) = (Saccept, _, L)
  - t(S5, 0) = (Sreject, 0, L)
  - t(S5, 1) = (Sreject, 1, L)
  - t(S5, x) cannot happen

Moral of the story

The transition mapping is a very inefficient way of specifying a Turing machine, as a lot of transitions cannot occur, unlike what we saw for a finite state acceptor, where the input alphabet was exactly the alphabet of the language. Here, the input alphabet is just a subset of the tape alphabet ($A \Subset \tilde{A}$).
Therefore we will specify a turing machine via either an algorithm or the transition diagram only, avoiding transition mapping.

Defining recognized languages

To figure out which languages are recognized by a turing machine, we need to introduce the notion of a *configuration*.
As a Turing machine goes through computations, changes take place in

1. The state of the machine
2. The tape contents
3. The tape head location

A setting of these three items is called a *configuration*.

Representing configurations

We represent a configuration as u, s_i, v where u and v are strings in the tape alphabet $\tilde{A}$ and s_i is the current state of the machine. The tape contexts are the string uv and the current location of the tape head is on the first character/Symbol of V. The assumption is that the tape only contains blanks after the last symbol in v.