Today start Turing machines:

Chapter 3 of [Sip].

DFA is \((Q, \Sigma, \delta, q_0, F)\)

TM is \((Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej}, \lambda)\)

or simply:

\((Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})\)
Idea: a Turing machine has a tape

Cell contains a "tape symbol":

Tape alphabet:

- contains \( \Sigma \),
- contains blank symbol \( \Lambda \)
- contains any other symbols
you want

\[ M = \text{Turing machine}, \text{ over } \Sigma = \{a, b\} \]
on input \( i = abba \in \Sigma^* \)
then initially the tape looks like

\[
\begin{array}{cccccccccc}
\downarrow & a & b & b & a & u & u & u & u & \cdots \\
\end{array}
\]
tape head - can move right but also to the left

tape is a read/write tape

Mechanics:

\[
\begin{array}{cccccccccccccccccc}
\downarrow & \cdots & b & b & u & b & a & b & a & u & u & u & \cdots \\
\end{array}
\]
\( \delta \) function tells us: (1) what is
(2) do we move L or R

Q: What happens when you get an instruction to move L = left when

\[ \begin{array}{c}
\downarrow \\
1 \quad 1 
\end{array} \]

Let's say (most standard): in this case you don't move, tape head remains over cell 1.

Q: What about \( S = \) stay still versus L, R

For now (but not for 2-tape or multi-tape TM) we don't
allow $S$, on $L, R$

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

Formally, a Turing machine is a 7-tuple

$$(Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$$

- Set of input tape alphabet
- Set of states

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

intuitively

$$\delta (q, \gamma) = (q', \gamma', L \text{ or } R)$$

$q'$ is new state, $\gamma'$ symbol written over $\gamma$ where $\nabla$ is situated
Any $T.M.$, on input $i \in \Sigma^*$ either accepts $i$ — you transition to $q_{acc}$ or rejects $i$ — in $q_{rej}$ loops — doesn't halts, i.e. never reaches $q_{acc}, q_{rej}$.

A $T.M., M$, recognizes the language

$$\{ i \in \Sigma^* \mid M \text{ accepts } i \}$$

A $T.M., M$, is a decider if it always halts, i.e. for any $i \in \Sigma^*$, $M$ either accepts or rejects $i$. 

$q_{accept}, q_{reject}$ states s.t. when you reach them, the $T.M.$ halts.

Any $T.M.$, on input $i \in \Sigma^*$ either

accepts $i$ — you transition to $q_{acc}$
rejects $i$ — in $q_{rej}$ loops — doesn’t halts, i.e.

never reaches $q_{acc}, q_{rej}$.
A language \( L \subseteq \Sigma^* \) is (Turing-)recognizable if some \( T.M., M \), recognizes \( L \), and is (Turing-)decidable if some \( T.M., M \), that is a decider recognizes \( L \).

Remark: Unlike Python (C, APL, --) programs, T. M.'s have only a single accept state, \( Q_{\text{acc}} \), and a single
reject state

\[ L = \{ w \in \{a, b\}^* \mid w \text{ ends in } \] an "a" \}

DFA:

\[ \text{AccFut}(E) = \text{AccFut}(b) \] and therefore also

\[ \text{AccFut}(a) = \text{AccFut}(b) \]
T.M. (technically, although qacc + qrej, qo can be one of qacc | qrej)

This indicates tape head is over cell 1

Tape alphabet tells you what state you're in

OR qabba on qabba...