Turing Machines

David Robinson

Turing Machine

A **Turing Machine** is like a finite automaton with an infinite tape that is used as memory. At every step, a Turing Machine can:

- Transition based on the current state and the tape symbol at the current position, like a DFA
- Write a symbol to the tape at the current position if it wishes
- Move its head either left or right on the tape

A Turing Machine is defined by $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ where:

- Q is a finite set of states
- Σ is the input alphabet, not containing the blank symbol $\square$
- Γ is the tape alphabet, a superset of Σ and contains $\square$
- q_0 is the start state
- q_{accept} is the accept state
- q_{reject} is the reject state, where $q_{\text{accept}} \neq q_{\text{reject}}$

An **enumerator** is a Turing machine with a printer. Instead of accepting input, it generates all strings accepted by its language.

A **multitape Turing machine** has a finite k number of tapes, each with its own read-write head.

$$\delta : Q \times \Gamma^k \to Q \times \Gamma^k \times \{L, R\}^k$$

A **nondeterministic Turing machine** has a finite number of nondeterministic choices.

$$\delta : Q \times \Gamma \to \mathbf{P}(Q \times \Gamma \times \{L, R\})$$