If the alphabet is infinite, a DFSA can have infinite computation. However, it cannot if the alphabet is finite.

2.1.3
(a)

(b)
2.1.5
(a)
(i)

(ii)

(iii)
(iv)

(b) Define a deterministic 2-tape finite automaton to be a sextuple \((K_1, K_2, \Sigma, \delta, s, F)\), where

- \(K_1\) and \(K_2\) are finite and disjoint sets of states,
- \(\Sigma\) is an alphabet,
- \(s \in K_1 \cup K_2\) is the initial state,
- \(F \subseteq K_1 \cup K_2\) is the set of final states,
- \(\delta\) is a function from \((K_1 \cup K_2) \times \Sigma\) to \((K_1 \cup K_2)\).

A configuration for such a machine is an ordered triple \((q, w_1, w_2)\), where

- \(q \in K_1 \cup K_2\) is the current state,
- \(w_1 \in \Sigma^*\) is the remaining input on the first tape, and
- \(w_2 \in \Sigma^*\) is the remaining input on the second tape.

We define \(\vdash_M\) to be the reflexive transitive closure of \(\vdash\).

We say that \(M\) accepts an ordered pair of strings \((w_1, w_2)\), \(w_1, w_2 \in \Sigma^*\), if \((s, w_1, w_2) \vdash_M^* (q, e, e)\) for some \(q \in F\).

Finally, if \(L \subseteq \Sigma^* \times \Sigma^*\) is a set of ordered pairs of strings, we say that \(M\) accepts \(L\) if \(M\) accepts \((u, w)\) if and only if \((u, w) \in L\).
2.2.9
(a)

(b)