L_1 = \{ x \in \Sigma^* : x \text{ begins with } a \text{bb} \}

Let \( \Sigma = \{a, b\} \). 

L_2 = \{ x \in \Sigma^* : x \text{ ends with } a \text{bb} \}

Design DFAs M_1 and M_2 s.t.

\[ L(M_1) = L_1 \text{ and } L(M_2) = L_2. \]

M_1 has 5 states, 10 arcs, and is a DFA s.t. \( L(M_1) \cong L_1 \).

\[ \text{Nirvana State} \]

Mirror Image Strategy for M_2

This diagram correctly denotes valid computation paths:

\[ \Gamma = (0, a, 0, a, 0, a, 1, b, 2, b, 3) \text{ next char is } a. \]

We cannot process it, so this cannot be completed to a valid computation on x.

Try again: \[ \Gamma' = (0, a, 0, a, 0, b, 0, b, 0, a, 1, b, 2, b, 3) \] OK since we processed all of x.

Thus \( \Gamma' \) is a valid accepting computation of M_2 on x, so x \( \epsilon L_2 \). And

\[ L(M_2) = L_2. \]

Example: x = aabbabbb

Defn: An NFA (without \( \epsilon \)-Arcs) has the same "class object definition" as a DFA, for

\[ L_2 \text{ with } S \subseteq Q \times \Sigma \times Q \text{ without requiring } \delta \text{ to be a function from } Q \times \Sigma \text{ to } Q \]

This is a legal DFA.

It is correct: \( L(M_2) = L_2 \).

It has no dead state and no "nirvana" state.

If we had \( \delta_2(0, b) = 0 \), then the resulting DFA M_2' would accept y = aabbabbb, but y \( \not\in L_2 \), so M_2' is unsound.
Our NFAs: \[ N_1 = \begin{array}{c l} \alpha & \beta \rightarrow a & b & \\ \alpha & b \rightarrow \beta \end{array} \]

They suggest shorthand for \( L_1 \) and \( L_2 \).

\[ L_1 = a \cdot b \cdot b \cdot (a \cup b)^* \]

Begins with a b b followed by zero or more occurrences of \( a \cup b \) which can each be \( a \) or \( b \).

\[ L_2 = (a \cup b)^* \cdot a \cdot b \cdot b \]

What language does this regular expression denote?

\[ L(R_2) = \{ x \in \{a, b\}^* \mid x \text{ has abb as a substring} \} \]

Now consider \( R_3 = (a \cup b)^* \cdot a \cdot b \cdot b \).

\[ L(R_3) \text{ includes } L_1 \cup L_2 \text{ and other strings like } z = aabba. \]

How about \( R_4 = (a \cup b)^* \cdot Lba \)?

We can diagram a corresponding NFA, using an \( \varepsilon \)-arc.

\[ N_4 \]

Define: An NFA (with \( \varepsilon \)-arcs) is a 5-tuple \( N = (Q, \Sigma, \delta, s, F) \) with \( s \in Q \) and \( F \subseteq Q \) as before, but now \( \delta \subseteq Q \times (\Sigma \cup \{\varepsilon\}) \times Q \).

\[ N_4 \text{ has } s = 0, F = \{3\}, \text{ and } \delta = \{(1, a, 1), (3, b, 1), (1, \varepsilon, 2), (2, b, 4), (1, a, 2)\}. \]

Computations similarly liberalize:

\[ \vdash (1, a, 3, b, 1, \varepsilon, 2, b, 4, a, 2) \text{ is a valid computation that processes } a \cdot b \cdot \varepsilon \cdot b \cdot a = aabba \text{ from state 1 to state 2.} \]

\[ L(N_4) = L(R_4) \text{ as required.} \]