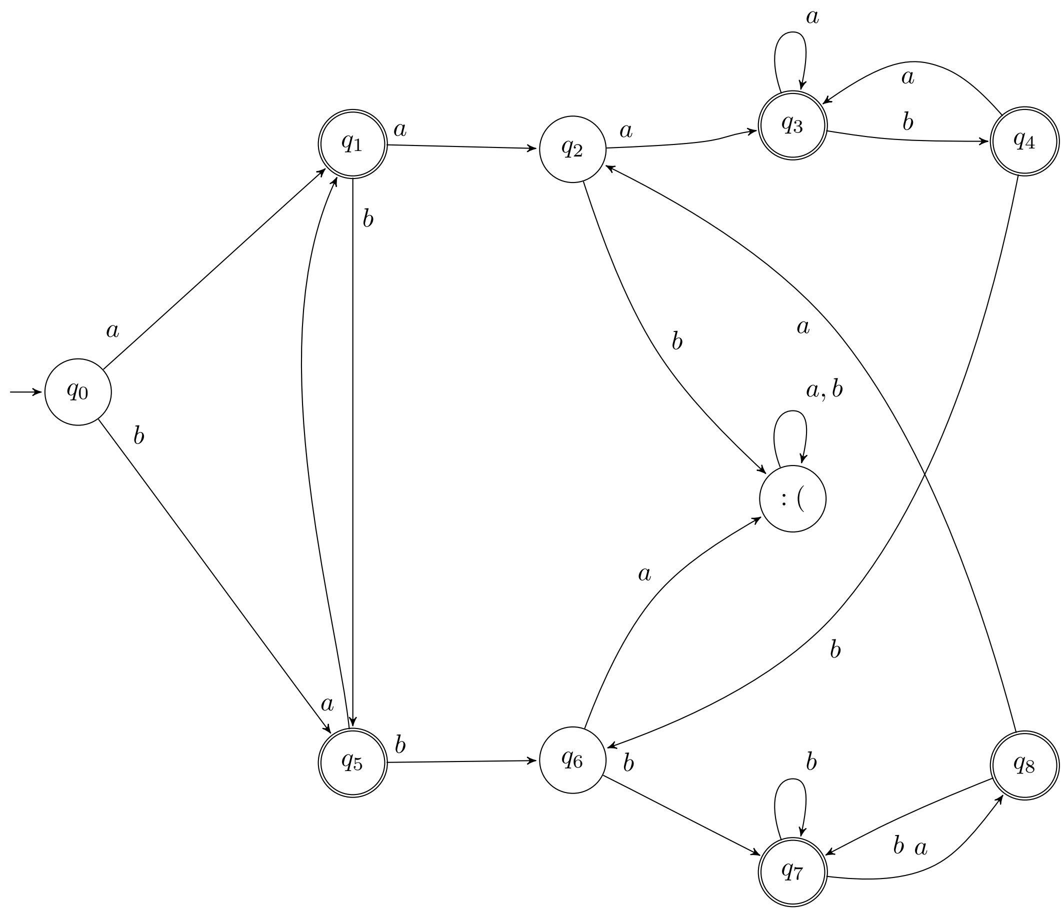
Administrativia: no discussions, no extra material consulted

1 Problem



,

Consider the two-state DFA M shown in the figure above.

Let $L = \{\{a, b\}^* \mid$ there is no run of length 2 $\}$.

Let

$$P(x) : \delta^*(q_0, x) = \begin{cases} q_1 \text{ IMPLIES} & x \in \mathfrak{L}(b\{ab\}^*a + a\{ba\}^* = P_1) \\ q_3 \text{ IMPLIES} & x \in \mathfrak{L}((b\{ab\}^*a + a\{ba\}^*)aa\{a\}^* + \\ & \big((a\{ba\}^*b + b\{ba\}^*b)b\{b\}^*a\{ba\}^*aa\{a\}^*\big) \cdot \\ & (\lambda + b\{ab\}^*b\{b\}^*a\{ba\}^*aa\{a\}^*) = P_3) \\ q_4 \text{ IMPLIES} & x \in \mathfrak{L}(P_3 b\{ba\}^* = P_4) \\ q_5 \text{ IMPLIES} & x \in \mathfrak{L}(a\{ba\}^*b + b\{ba\}^* = P_5) \\ q_7 \text{ IMPLIES} & x \in \mathfrak{L}((P_5 + P_4)(bb\{b\}^*) \\ & (\lambda + \{ab\}^*(\lambda + aa\{a\}^*b\{ab\}^*bb\{b\}^*)) = P_7) \\ q_8 \text{ IMPLIES} & x \in \mathfrak{L}(P_7 a\{ba\}^*) \end{cases} \tag{1}$$

Lemma 1.1

M accepts the language L.

Proof.

We use structural induction on $x \in L$.

Base Case

If $x = \lambda$, then there are zero characters in x and hen there are no ru. Thus, the claim holds in case $x = \lambda$.

Inductive Step

Suppose the claim holds for $x = y$ and consider the case when $x = yt$ for some Σ^* and $t \in \Sigma$, where $\Sigma = \{a, b\}$.

There are two cases: $t = a$ and $t = b$.

If $t = a$, then $\delta^*(q_0, ya) = \delta(\delta^*(q_0, y), a)$ by definition of δ^*.

Looking at the mess of 8 cases and the diagram and considering the inductive hypothesis, we see that none of the transitions introduces a run of length 2, and hence the claim holds if $t = a$.

If $t = b$, then $\delta^*(q_0, yb) = \delta(\delta^*(q_0, y), b)$ by definition of δ^*, and the similar reasoning holds.

Conclusion

By strong induction, M accepts the language L.

□

To see why the regular expressions which define the languages in the equation (1) are indeed correct, the reader is referred to the diagram, even though the author fully admits that this reasoning in no way constitutes a proof.

The main intuition is that the nodes q_0, q_1, q_2, q_5 and q_6 *sort* the input string in such a way that it does not have a run of length 2 at the points defined by the first few regex terms. Then the process *goes into the loop* by first allowing the runs of length greater than three and then ensuring that there are no runs of length 2, which is achieved by linking from q_4 and q_8 to the respective *run-2 trimmers* (q_6 and q_2) as defined by the edges after the nodes q_2 and q_6, which then continue into the loop which consists of allowed states.