Defining a Language from a DFA

To define a language from a given DFA (Deterministic Finite Automaton), you describe the set of strings that the DFA accepts. Here's a step-by-step guide:

1. Describe the DFA

A DFA is typically defined by a 5-tuple $M = (Q, \Sigma, \delta, q_0, F)$, where:

- Q: A finite set of states.
- Σ: The input alphabet (a finite set of symbols).
- δ: The transition function $\delta : Q \times \Sigma \to Q$, which determines the state transitions based on the current state and input symbol.
- q_0: The start state, where $q_0 \in Q$.
- F: The set of accept (final) states, where $F \subseteq Q$.

2. Define the Language $L(M)$

The language recognized (accepted) by the DFA M, denoted as $L(M)$, is the set of all strings over the alphabet Σ that the DFA accepts.

Formally, $L(M)$ is defined as:

$$L(M) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$$

where:

- w is a string over the alphabet Σ.
- Σ^* is the set of all strings (including the empty string) over the alphabet Σ.
- $\hat{\delta}$ is the extended transition function, which is recursively defined as:

$$\begin{aligned} \hat{\delta}(q, \epsilon) &= q && \text{(for the empty string } \epsilon\text{)} \\ \hat{\delta}(q, aw) &= \delta(\hat{\delta}(q, a), w) && \text{(for } a \in \Sigma \text{ and } w \in \Sigma^*\text{)} \end{aligned}$$

3. Interpretation

- The language $L(M)$ consists of all strings that, when processed by the DFA starting from the start state q_0, lead to one of the accept states in F.
- If a string w is in $L(M)$, the DFA can process w and end in an accept state.
- If a string w is not in $L(M)$, the DFA either ends in a non-accepting state or cannot process the string according to its transition rules.

Example

Consider a DFA $M = (Q, \Sigma, \delta, q_0, F)$ with:

- $Q = \{q_0, q_1, q_2\}$
- $\Sigma = \{0, 1\}$
- δ defined as:

$$\begin{aligned}
\delta(q_0, 0) &= q_0 \\
\delta(q_0, 1) &= q_1 \\
\delta(q_1, 0) &= q_2 \\
\delta(q_1, 1) &= q_0 \\
\delta(q_2, 0) &= q_1 \\
\delta(q_2, 1) &= q_2
\end{aligned}$$

- q_0 is the start state.
- $F = \{q_2\}$.

Language $L(M)$:

- The DFA accepts a string if it ends in state q_2.
- $L(M)$ is the set of strings over $\{0, 1\}$ that result in the DFA being in state q_2 after reading the entire string.

Conclusion

To define a language from a DFA, you describe the set of all strings that the DFA accepts. This set is the language recognized by the DFA, and it is formally defined using the extended transition function $\hat{\delta}$ that describes how the DFA processes each string.

Finite Automata: Language Accepted by DFA

Contents

Here we are going to formally define what is meant by a DFA (Deterministic Finite Automaton) accepting a string or a language.

Acceptance of a String by a DFA

A string w is accepted by a DFA $M = \langle Q, \Sigma, q_0, \delta, A \rangle$ if and only if $\hat{\delta}(q_0, w) \in A$. That is, a string is accepted by a DFA if and only if the DFA starting at the initial state ends in an accepting state after reading the string.

Acceptance of a Language by a DFA

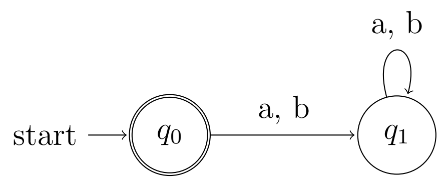
A language L is accepted by a DFA $M = \langle Q, \Sigma, q_0, \delta, A \rangle$ if and only if

$$L = \{ w \mid \hat{\delta}(q_0, w) \in A \}$$

That is, the language accepted by a DFA is the set of strings accepted by the DFA.

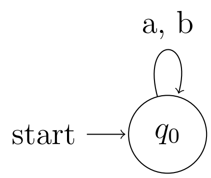
Examples

Example 1



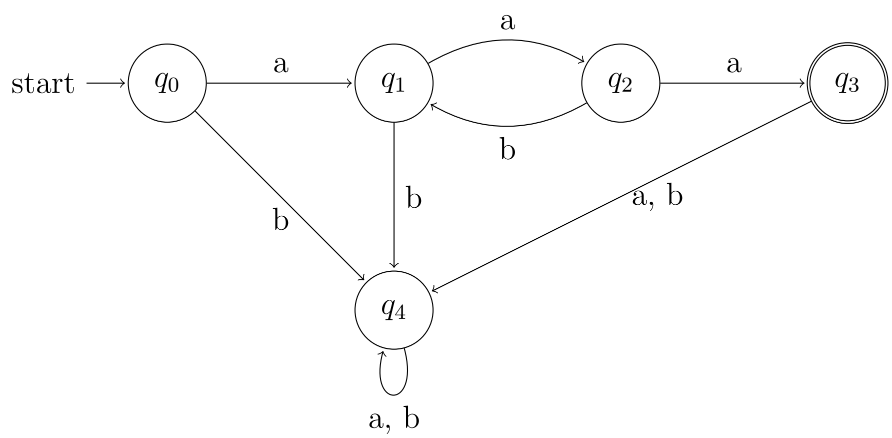
This DFA accepts $\{\epsilon\}$ because it can go from the initial state to the accepting state (also the initial state) without reading any symbol of the alphabet, i.e., by reading an empty string ϵ. It accepts nothing else because any non-empty symbol would take it to state 1, which is not an accepting state, and it stays there.

Example 2



This DFA does not accept any string because it has no accepting state. Thus, the language it accepts is the empty set $\emptyset$.

Example 3: DFA with One Cycle



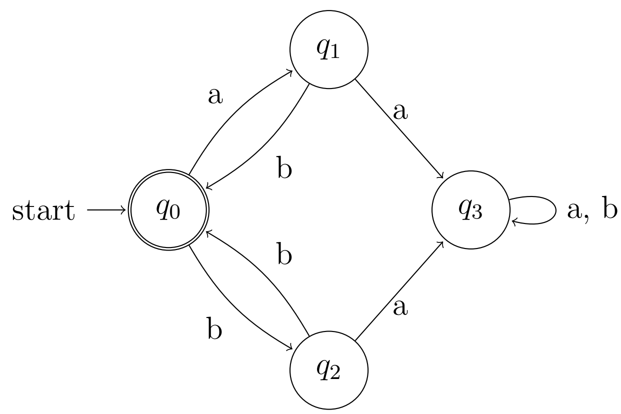
This DFA has a cycle: 1 - 2 - 1, and it can go through this cycle any number of times by reading the substring "ab" repeatedly.

To find the language it accepts:

- First, from the initial state go to state 1 by reading one "a".
- Then from state 1, go through the cycle 1 - 2 - 1 any number of times by reading the substring "ab" any number of times to come back to state 1. This is represented by $(ab)^*$.
- Finally, from state 1 go to state 2 and then to state 3 by reading "aa".

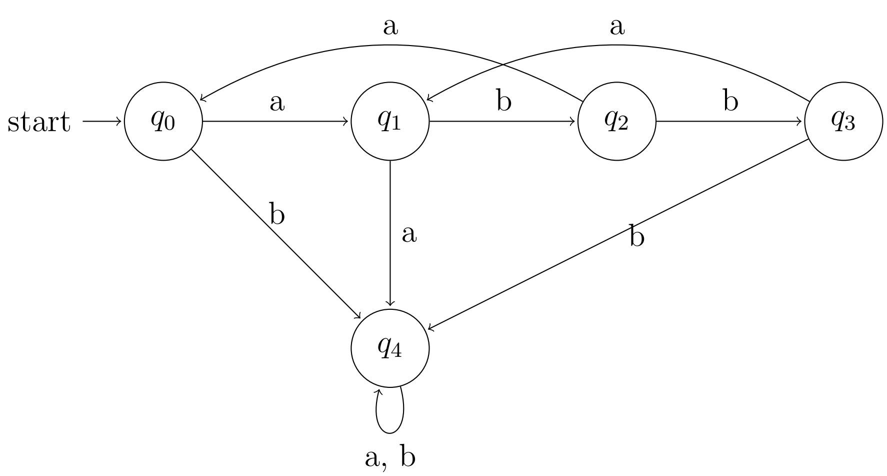
Thus, a string that is accepted by this DFA can be represented by $a(ab)^*aa$.

Example 4: DFA with Two Independent Cycles



This DFA has two independent cycles: 0 - 1 - 0 and 0 - 2 - 0, and it can move through these cycles any number of times in any order to reach the accepting state from the initial state, such as 0 - 1 - 0 - 2 - 0 - 2 - 0. Thus, a string that is accepted by this DFA can be represented by $(ab + bb)^*$.

Example 5: DFA with Two Interleaved Cycles



This DFA has two cycles: 1 - 2 - 0 - 1 and 1 - 2 - 3 - 1.

To find the language accepted by this DFA:

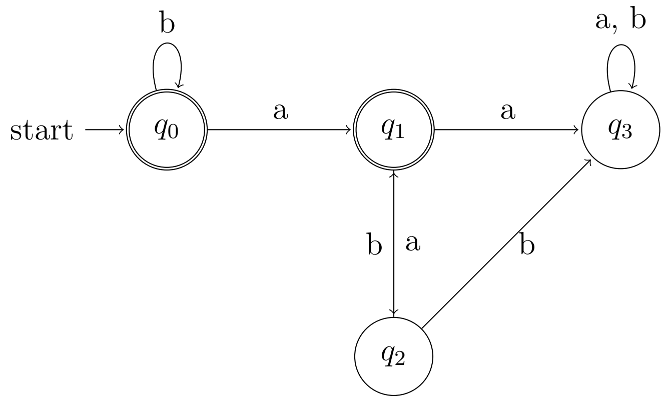
- First, from state 0 go to state 1 by reading ‘a‘. (Any other state which is common to these cycles such as state 2 can also be used instead of state 1).

- Then from state 1, go through the two cycles 1 - 2 - 0 - 1 and 1 - 2 - 3 - 1 any number of times in any order by reading substrings ”baa” and ”bba”, respectively. At this point, a substring $a(baa + bba)^*$ will have been read.

- Then go from state 1 to state 2 and then to state 3 by reading ”bb”.

Thus, altogether $a(baa + bba)^*bb$ will have been read when state 3 is reached from state 0.

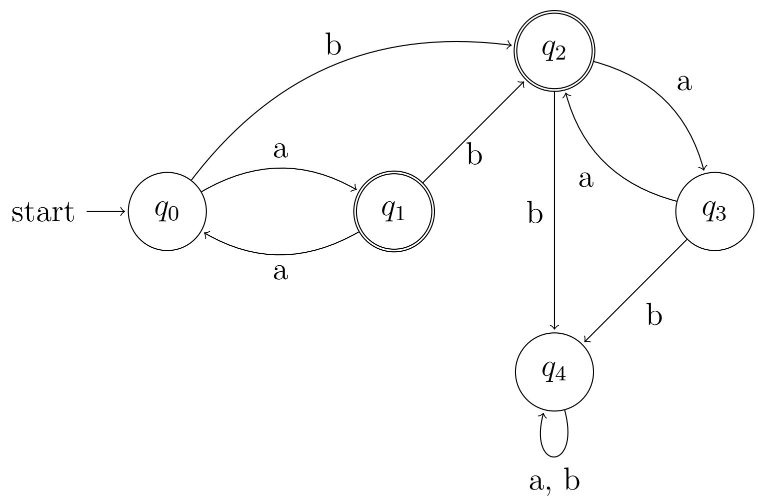
Example 6



This DFA has two accepting states: 0 and 1. Thus, the language that is accepted by this DFA is the union of the language accepted at state 0 and the one accepted at state 1.

- The language accepted at state 0 is b^*.
- To find the language accepted at state 1, first at state 0 read any number of "b"s. Then go to state 1 by reading one "a". At this point (b^*a) will have been read. At state 1, go through the cycle 1 - 2 - 1 any number of times by reading the substring "ba" repeatedly. Thus, the language accepted at state 1 is $b^*a(ba)^*$.

More examples: Tut 1 Q5



This DFA contains 2 accepting states, suggesting the union of the language accepted at state 1 and state 2.

- The language accepted at state 1 is
  - Case 1-1: $a(aa)^*$.
- The language accepted at state 2 can be
  - Case 2-1: $(aa)^*b$: ((0 - 1 - 0)* - 2) ($Case\ 2-1 \subseteq Case\ 2-3$)
  - Case 2-2: $(aa)^*b(aa)^*$: ((0 - 1 - 0)* - (2 - 3 - 2)*) ($Case\ 2-2 \subseteq Case\ 2-4$)
  - Case 2-3: $a^*(aa)^*b$: (0 - (1 - 0 - 1)* - 2) ($Case\ 2-3 \subseteq Case\ 2-2$)
  - Case 2-4: $a^*(aa)^*b(aa)^*$: (0 - (1 - 0 - 1)* - (2 - 3 - 2)*)
  - Case 2-5: $b(aa)^*$: (0 - (2 - 3 - 2)*) ($Case\ 2-5 \subseteq Case\ 2-2$)

All unioned together, language accepted by this DFA is

$$L = a(aa)^* \cup a^*(aa)^*b(aa)^*$$