Daily Problem: 23–Jan–2026

Problem Statement

1. Show that every NFA with possibly multiple accepting states can be transformed into an equivalent NFA with exactly *one* accepting state.
2. Show that every NFA with possibly multiple initial states can be transformed into an equivalent NFA with exactly *one* initial state.

NFA Definition

An NFA is a tuple

$$N = (Q, \Sigma, \delta, I, F),$$

where

- Q is a finite set of states,
- Σ is the input alphabet,
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ is the transition function,
- $I \subseteq Q$ is the set of *initial* states (possibly of size more than 1),
- $F \subseteq Q$ is the set of *accepting* states (possibly of size more than 1).

Part 1: One Accepting State

Goal

Given an NFA $N = (Q, \Sigma, \delta, I, F)$ with $|F| \geq 1$, construct an equivalent NFA $N' = (Q', \Sigma, \delta', I', F')$ having exactly one accepting state.

Construction

Let $f_{\text{new}} \notin Q$ be a fresh new state. Define:

$$Q' = Q \cup \{f_{\text{new}}\},$$

$$I' = I,$$

$$F' = \{f_{\text{new}}\}.$$

Define the transition function δ' as follows:

1. For every $q \in Q \setminus F$ and every $a \in \Sigma$, let:

   $$\delta'(q, a) = \delta(q, a).$$

2. For every original accepting state $f \in F$ and every $a \in \Sigma$, define:

   $$\delta'(f, a) = \emptyset,$$

   and introduce an ε-transition from f to f_{new}:

   $$\delta'(f, \varepsilon) = \{f_{\text{new}}\}.$$

3. For the new accepting state:

   $$\delta'(f_{\text{new}}, a) = \emptyset \quad \text{for all } a \in \Sigma.$$

Correctness Argument

If a string w is accepted by the original NFA, there exists a path from some initial state to some accepting $f \in F$ while reading w. In the constructed NFA N', the same path exists and ends in f, followed by an ε-transition to the unique accepting state f_{new}. Hence w is accepted by N'.

Conversely, if N' accepts w, the accepting run must end in f_{new}, which is only reachable by an ε-transition from some $f \in F$. Thus w must have taken the NFA from an initial state to some $f \in F$ in the original automaton. Therefore the original NFA accepts w.

Thus the two NFAs are equivalent and N' has exactly one accepting state.

Conclusion for Part 1

Every NFA with multiple accepting states can be transformed into an equivalent NFA with exactly one accepting state, by adding one new accepting state and redirecting all old accepting states to it via ε-moves.

Part 2: One Initial State

Goal

Given an NFA $N = (Q, \Sigma, \delta, I, F)$ with possibly multiple initial states I, construct an equivalent NFA with exactly one initial state.

Construction

Let $q_{\text{new}} \notin Q$ be a new state. Define:

$$Q' = Q \cup \{q_{\text{new}}\},$$

$$I' = \{q_{\text{new}}\},$$
$$F' = F.$$

Define δ' as follows:

- For all $q \in Q$ and $a \in \Sigma$,

$$\delta'(q, a) = \delta(q, a).$$

- For the new initial state,

$$\delta'(q_{\text{new}}, \varepsilon) = I,$$

  meaning that the new initial state can nondeterministically jump via an ε-transition into any of the original initial states.

Correctness Argument

If the original NFA accepts w, there exists an initial state $i \in I$ and a path from i to some accepting state on input w. In the new NFA, we can take the ε-transition from q_{new} to i and then follow the same path, accepting w.

Conversely, if the new NFA accepts w, the accepting run must begin at q_{new}, immediately take an ε-transition to some $i \in I$, and then follow a valid accepting run of the original NFA. Hence w is accepted by the original NFA.

Therefore both NFAs recognize the same language.

Conclusion for Part 2

Every NFA with multiple initial states can be transformed into an equivalent NFA with exactly one initial state, by adding a fresh initial state with ε-transitions to all original initial states.

Final Conclusion

We have shown:

1. Any NFA with multiple accepting states can be converted into an equivalent NFA with exactly one accepting state.

2. Any NFA with multiple initial states can be converted into an equivalent NFA with exactly one initial state.

Both transformations preserve the recognized language and only require adding one fresh state with appropriate ε-transitions.