Automaton State Minimization and AST Optimization

Nnamdi Michale Okpala

11/12/2024

1 Layman's Explanation

Automaton state minimization is about taking a finite state machine (FSM)—which may have redundant states—and simplifying it to use the smallest number of states possible, while still performing the same behavior. This concept is particularly relevant when associated with minimizing the **abstract syntax tree (AST)**, which is a tree representation of the machine's rules or transitions.

2 Formal Definition

2.1 Automaton Representation

Let the automaton A be represented as a 5-tuple:

$$A = (Q, \Sigma, \delta, q_0, F)$$

where:

- Q: Finite set of states
- Σ: Finite alphabet (input symbols)
- $\delta : Q \times \Sigma \to Q$: Transition function
- $q_0 \in Q$: Initial state
- $F \subseteq Q$: Set of accepting (final) states

2.2 Abstract Syntax Tree (AST)

The AST represents the structure of transitions and states in the automaton as a tree. Each node corresponds to:

- A state $q \in Q$
- An input $\sigma \in \Sigma$ or output transition $\delta(q, \sigma)$

2.3 State Equivalence

Define two states $p, q \in Q$ as **equivalent** ($p \sim q$) if for every possible input sequence $w \in \Sigma^*$, the automaton starting at p and q ends in the same type of state (both accepting or both non-accepting).

Mathematically:

$$p \sim q \iff \forall w \in \Sigma^*, \delta^*(p, w) \in F \iff \delta^*(q, w) \in F$$

Here, δ^* is the extended transition function for sequences:

$$\delta^*(q, \epsilon) = q, \quad \delta^*(q, aw) = \delta^*(\delta(q, a), w)$$

2.4 Minimization

The minimization process constructs a new automaton $A' = (Q', \Sigma, \delta', q_0', F')$ where:

- Q': Partition of Q into equivalence classes under $\sim$
- $\delta'(C, a) = [\delta(q, a)]$ for any $q \in C$, where C is an equivalence class
- $q_0' = [q_0]$ (the equivalence class containing the initial state)
- $F' = \{C \in Q' \mid C \cap F \neq \emptyset\}$

3 Summary

To minimize an automaton:

1. Identify equivalent states using the equivalence relation $\sim$
2. Partition states into equivalence classes
3. Construct the reduced automaton using the partitions
4. Reflect the minimized structure in the AST by reducing nodes and transitions accordingly