1 Finite state machine minimization

- *Algorithm:* Finite state machine minimization (algo. 1)
- *Input:* A finite state machine
- *Complexity:* $O(|\Sigma| \cdot |F|)$
- *Data structure compatibility:* Finite state machine (Deterministic finite automaton)
- *Common applications:* compilers, network protocols, theory of computation

Finite state machine minimization

Given a finite state machine, minimize the states.

Description

1. Definition of FSM

   The formal definition of a finite state machine (deterministic finite automaton) is

   $$M: \ (Q, \Sigma, \delta, q_0, F)$$

   (a) Q: finite set of states

   (b) Σ: finite set of input symbols

   (c) $\delta : Q \times \Sigma \rightarrow Q$: transition function

   (d) $q_0 \in Q$: initial state

   (e) $F \subseteq Q$: accept state

   The automaton will accept a string w if it starts at start state q_0, and given each character in w, the transition rule will transit state to state according to δ, and the final state shall halt at F states.

2. Input

   The input of the algorithm shall be a well-defined finite state machine. Any illegal input should not be considered.

3. Complexity

   The time complexity of this algorithm is defined by the cost of each iteration and iteration time. For the outer iteration, it will need at most $|\Sigma|$ iterations, and for the inner iteration, it will need around $|F|$ operations. Thus we will need $O(|\Sigma| \cdot |F|)$ complexity.

4. Application

In computer science field, finite state machine can be used to treat string. It can set up several states to check whether to accept an arrival of strings or not.

In digital circuits field, finite state machine can be used to depict the behavior of a certain circuit, and it could link the combinatoric circuits with a desired function.

Finite State Machine can also be used to depict a lot of behaviors in natural science or social science. For example, it could be applied to analyze the relation in different social characters, and to analyze how an social event is carried out.

5. Detailed Algorithm

This algorithm is introduced by Hopcroft. The idea behind is partition refinement, which means partition a large set into several small sets by their behavior.

At the very beginning, the are partitioned into two different groups, that is $\{F\}$ and $\{Q\backslash F\}$. These two groups are accepting states and rejecting states, obviously they are inequivalent.

Then it comes with the magic of this algorithm. That it separates $\{A\}$ with $\{W\backslash A\}$, where W initially to be F,

Algorithm 1: FSM minimization

Input : A formally defined FSM
Output: Minimized FSM

Function `MinimizeFSM`($M:\ (Q, \Sigma, \delta, q_0, F)$)**:**
  set $P \leftarrow \{F, Q\backslash F\}$;
  set $W \leftarrow \{F\}$;
  while $W \neq \varnothing$ **do**
    Choose $A \in W$;
    $W \leftarrow W\backslash A$;
    for s *in* Σ **do**
      $\mathrm{X} \leftarrow \{X : \delta(X \times c) \rightarrow A\}$; `/* In natural language, it means that X is a state that: the transition rule takes` c `in the state` X `will go to a state in set` A `*/`
      for $Y \subset P$ *s.t.* $X \cap Y \neq \varnothing$ **and** $Y\backslash X \neq \varnothing$ **do**
        if $|X \cap Y| \leq |Y\backslash X|$ **then**
          $\mathrm{W} \leftarrow \mathrm{W} \cup (X \cap Y)$;
        end if
        else
          $\mathrm{W} \leftarrow \mathrm{W} \cup (Y\backslash X)$;
        end if
      end for
    end for
  end while
end
return

References

- If available provide URLs, e.g. http://mywebsite.org
- Wikipedia is not acceptable if this is the unique reference
- Reference some books, or published articles
- Use reliable websites (no blog allowed) that are not likely to disappear any time soon