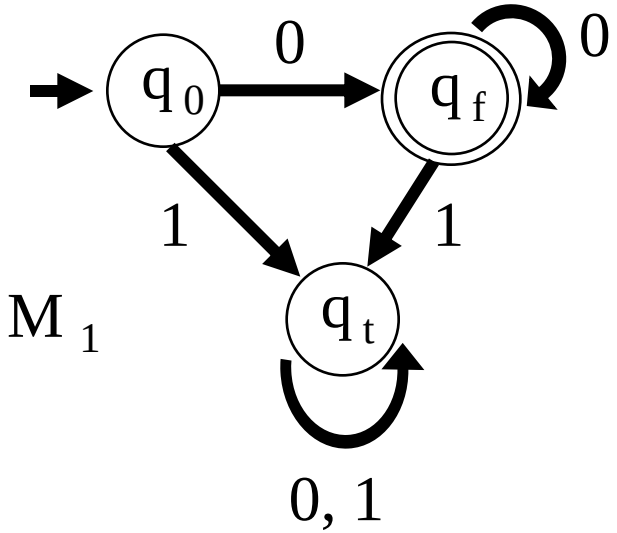
Minimization of Finite Automata

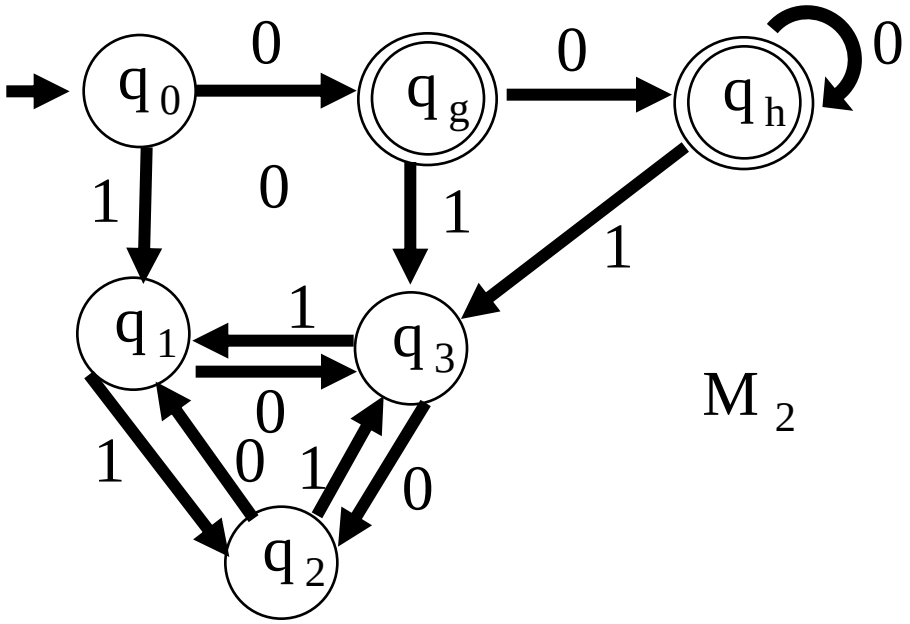
Minimization of Finite Automata

For a given regular language L, it is possible to construct infinitely many finite state automata to accept L. The reason is that based on a given DFA M_1, there would be a loop in the transition diagram. So we can construct a new DFA M_2 to have more states and $L(M_1) = L(M_2) = L$. And from DFA M_2, we can construct a DFA M_3 with even more states and $L(M_3) = L$.

Consider example 1, given a DFA M_1, we can construct a DFA M_2 to have more states and $L(M_1) = L(M_2) = L$.

Example 1: The following two DFA's M_1 and M_2 are equivalent. Both machines accept the set $\{0^n | n>0\}$ over the alphabet $\sum=\{0, 1\}$.





The states q_1, q_2, and q_3 of the machine M_2 are equivalent, we can combine these 3 states into one state as q_t in the machine M_1.

The states q_g and q_h of the machine M_2 are equivalent, we can combine these 2 states into one state as q_f in the machine M_1.

There are two classes of states that can be removed/merged from the original DFA without affecting the language it accepts to minimize it.

1. **Unreachable states** are those states that are not reachable from the initial state of the DFA, for any input string.

2**. Non-distinguishable states** are those that cannot be distinguished from one another for any input string.

DFA minimization is usually done in three steps, corresponding to the removal/merger of the relevant states. Since the elimination of non-distinguishable states is computationally the most expensive one, it is usually done as the last step.

.

Algorithms for minimizing Finite Automata

1.Hopcroft's algorithm

Based on partition refinement, partitioning the DFA states into groups by their behavior. These groups represent equivalence classes

whereby every two states of the same partition are equivalent if they have the same behavior for all the input sequences.

Moore's Algorithm

- Like Hopcroft's algorithm, it maintains a partition that starts off separating the accepting from the rejecting states, and repeatedly refines the partition until no more refinements can be made.

Brzozowski's Algorithm

- Reversing the edges of a DFA produces a non-deterministic finite automaton (NFA) for the reversal of the original language, and converting this NFA to a DFA using the standard powerset construction (constructing only the reachable states of the converted DFA) leads to a minimal DFA for the same reversed language. Repeating this reversal operation a second time produces a minimal DFA for the original language.

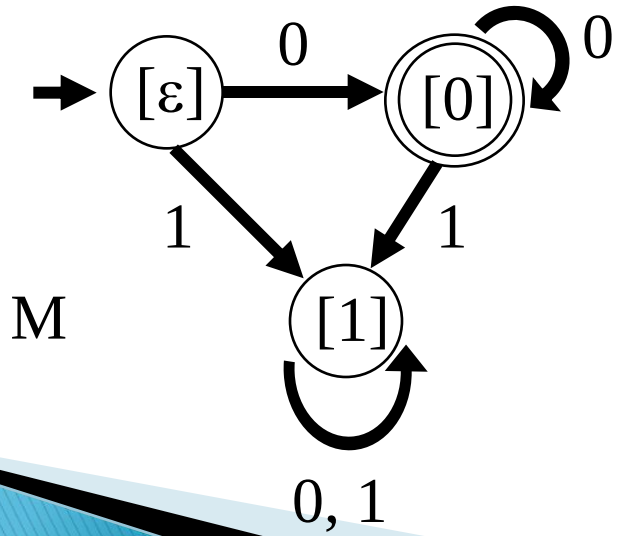
Example : Let L be a language over an alphabet $\sum = \{0, 1\}$, $L=\{ 0^n \mid n > 0\}$. Find a DFA M such that L(M)=L.

Solution :

The equivalence classes of R_L are :

$[0] = \{ 0^n \mid n > 0 \}$, $[\varepsilon] = \{\varepsilon\}$, $[1] = \sum^* \setminus \{0^n \mid n \geq 0 \}$.

By theorem 1, we have that the DFA M is as follows.



Thankyou