CS311 Formal Language and Automata

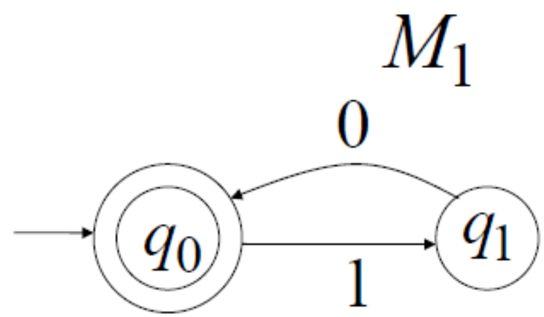
Tingting Chen
Computer Science
Cal Poly Pomona

NFA vs DFA

- Which one is more powerful?
- We can always find an equivalent DFA for any given NFA. We will see an example first, and then prove this statement.
- Therefore, NFA's are no more powerful than DFA's.

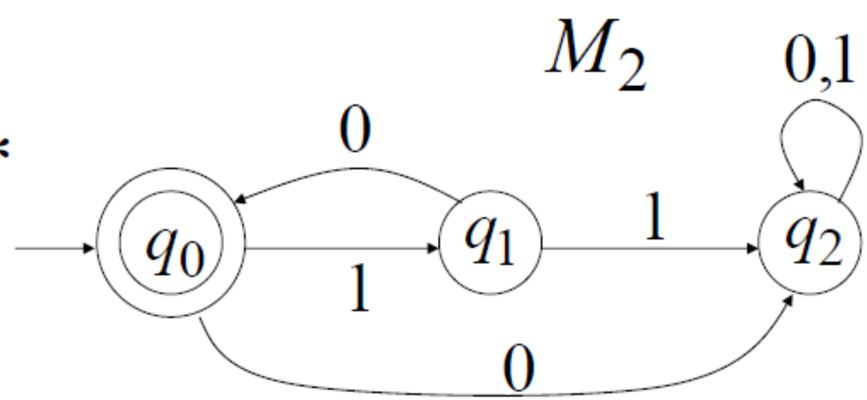
NFA:

$$L(M_1) = \{10\}^*$$



DFA:

$$L(M_2) = \{10\}^*$$



NFA vs DFA

- We are going to prove that the set of languages accepted by NFAs is the same set of languages accepted by DFAs.
- NFAs and DFAs have the same computation power.

- If for machine M_1 and machine M_2 $L(M_1)=L(M_2)$, then M_1 is **equivalent** to machine M_2

- Then it is equal to prove that
  - for any DFA, we can find its equivalent NFA;
    - {languages accepted by DFAs} $\subseteq$ { languages accepted by NFAs }
  - for any NFA, we can find its equivalent DFA.
    - {languages accepted by NFAs} $\subseteq$ { languages accepted by DFAs }

DFA -> NFA

- Every DFA is trivially an NFA.

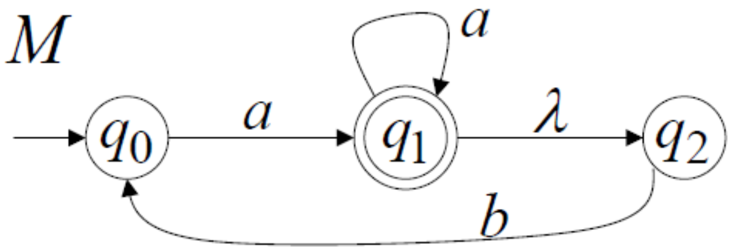
- For DFA $D = (Q, \Sigma, \delta_D, q_0, F)$, define an "equivalent" NFA $N = (Q, \Sigma, \delta_N , q_0, F)$ that has the exact same set of states, initial state and final states. Only difference is in the transition function.
  - $\delta_N (q, a) = \{\delta_D(q, a)\}$ for $a \in \Sigma$
  - $\delta_N (q, \lambda) = \{q\}$ for all $q \in Q$.

NFA ->DFA

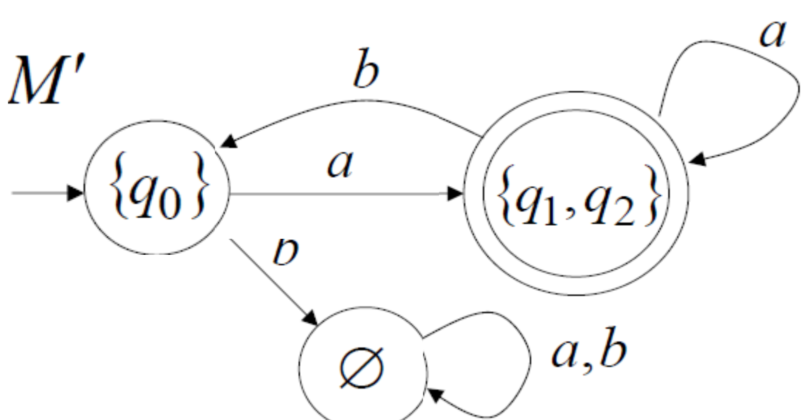
- We are given an NFA M.
- We want to convert it to an equivalent DFA M’, with L(M) = L(M’)

- Example: convert NFA to DFA

NFA:



DFA:



Formally, What we need to construct

- DFA $(Q, \Sigma, \delta_D, q_0, F)$

- If the NFA has states q_0, q_1, q_2,…,
- The DFA has states in the power set
  i.e., $\emptyset$, $\{q_0\}$, $\{q_1\}$, $\{q_1, q_2\}$, …

Procedure NFA to DFA

1 Initial state of NFA: q_0

→ Initial state of DFA: $\{q_0\}$

2 For every DFA's state $\{q_i, q_j, \ldots, q_m\}$

- Compute in the NFA

  $\delta^*(q_{i,}\ a)= \{q'_{i1}, q'_{i2}, \ldots q'_{in_i}\}$, $\delta^*(q_{j,}\ a)= \{q'_{j1}, q'_{j2}, \ldots q'_{jn_j}\}$,

  …,

  $\delta^*(q_{m,}\ a)= \{q'_{m1}, q'_{m2}, \ldots q'_{mn_m}\}$

- Add transition to DFA

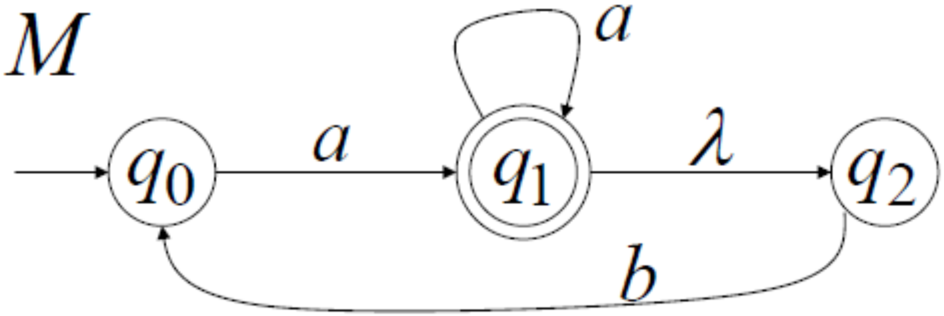
  $\delta(\{q_i, q_j, \ldots, q_m\}, a)$

  $=\{q'_{i1}, q'_{i2}, \ldots q'_{in_i,}\ q'_{j1}, q'_{j2}, \ldots q'_{jn_j,}\ \ldots, q'_{m1}, q'_{m2}, \ldots q'_{mn_m}\}$

Example

- NFA

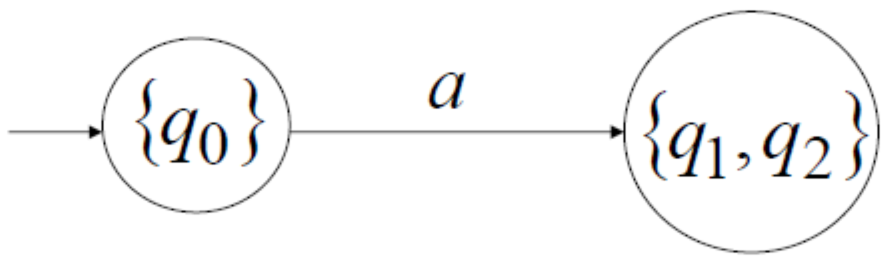
M



$$\delta^*(q_0, a) = \{q_1, q_2\}$$

- DFA

M'



$$\delta(\{q_0\}, a) = \{q_1, q_2\}$$

Procedure NFA to DFA

1 Initial state of NFA: q_0

→ Initial state of DFA: $\{q_0\}$

2 For every DFA's state $\{q_i, q_j, \ldots, q_m\}$

- Compute in the NFA

  $\delta^*(q_i, a) = \{q'_{i1}, q'_{i2}, \ldots q'_{in_i}\}$, $\delta^*(q_j, a) = \{q'_{j1}, q'_{j2}, \ldots q'_{jn_j}\}$,

  …,

  $\delta^*(q_m, a) = \{q'_{m1}, q'_{m2}, \ldots q'_{mn_m}\}$

- Add transition to DFA

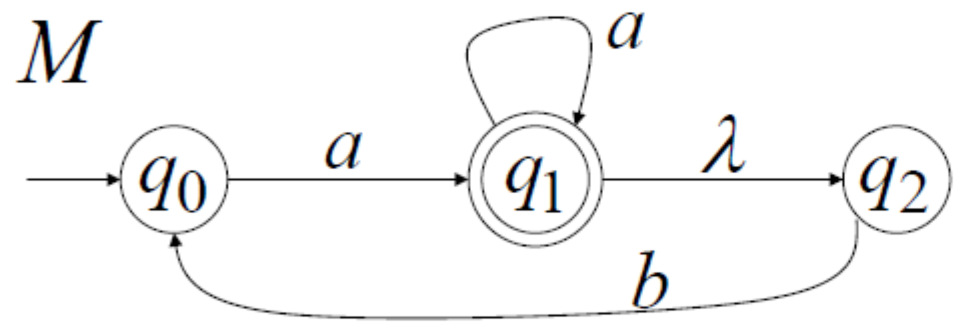
  $\delta(\{q_i, q_j, \ldots, q_m\}, a)$

  $= \{q'_{i1}, q'_{i2}, \ldots q'_{in_i}, q'_{j1}, q'_{j2}, \ldots q'_{jn_j}, \ldots, q'_{m1}, q'_{m2}, \ldots q'_{mn_m}\}$

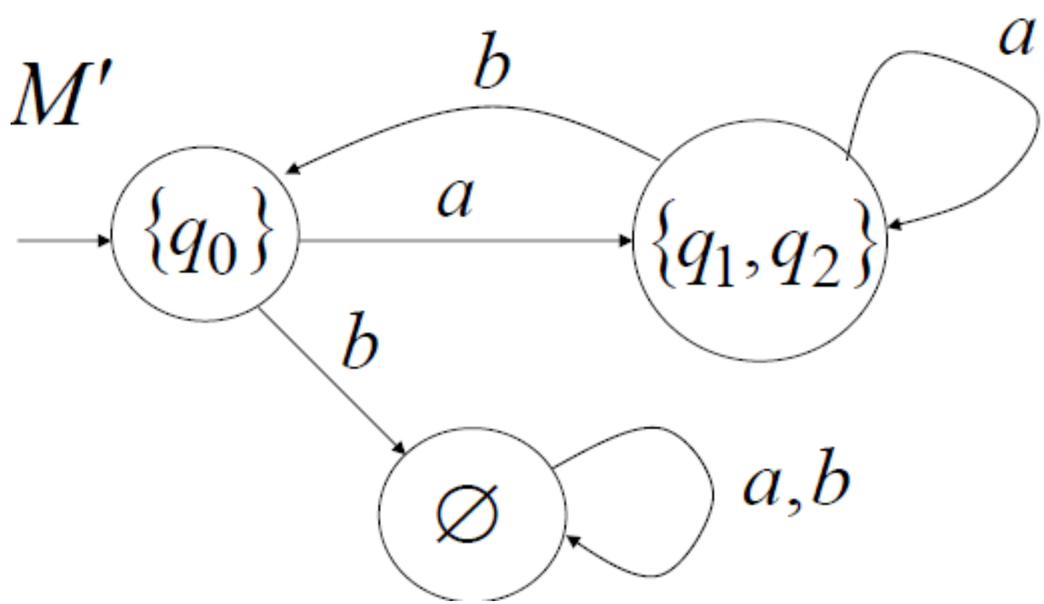
Repeat Step2 for all letters in alphabet, until no more transitions can be added.

Example

- NFA



- DFA



Procedure NFA to DFA

1 Initial state of NFA: q_0

$\rightarrow$ Initial state of DFA: $\{q_0\}$

2 For every DFA's state $\{q_i, q_j, \ldots, q_m\}$

- Compute in the NFA

  $\delta^*(q_{i,}\, a) = \{q'_{i1}, q'_{i2}, \ldots q'_{in_i}\}$, $\delta^*(q_{j,}\, a) = \{q'_{j1}, q'_{j2}, \ldots q'_{jn_j}\}$, …,

  $\delta^*(q_{m,}\, a) = \{q'_{m1}, q'_{m2}, \ldots q'_{mn_m}\}$

- Add transition to DFA

  $\delta(\{q_i, q_j, \ldots, q_m\}, a)$

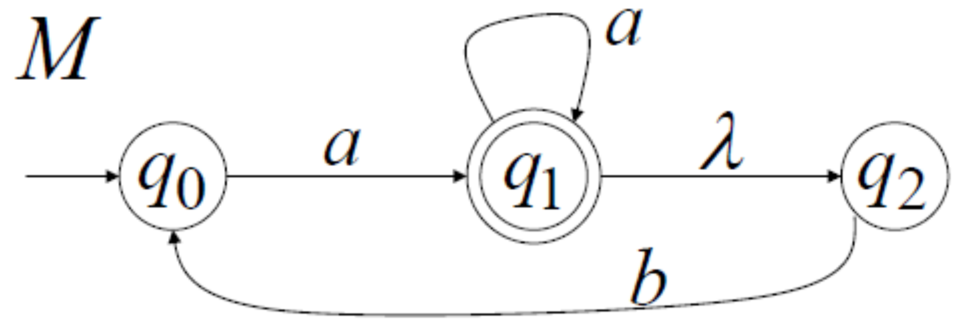
  $= \{q'_{i1}, q'_{i2}, \ldots q'_{in_i,}\; q'_{j1}, q'_{j2}, \ldots q'_{jn_j,}\; \ldots,\; q'_{m1}, q'_{m2}, \ldots q'_{mn_m}\}$

Repeat Step2 for all letters in alphabet, until no more transitions can be added.

3. For any DFA state $\{q_i, q_j, \ldots, q_m\}$, if q_j is accepting state in NFA Then $\{q_i, q_j, \ldots, q_m\}$ is accepting state in DFA.

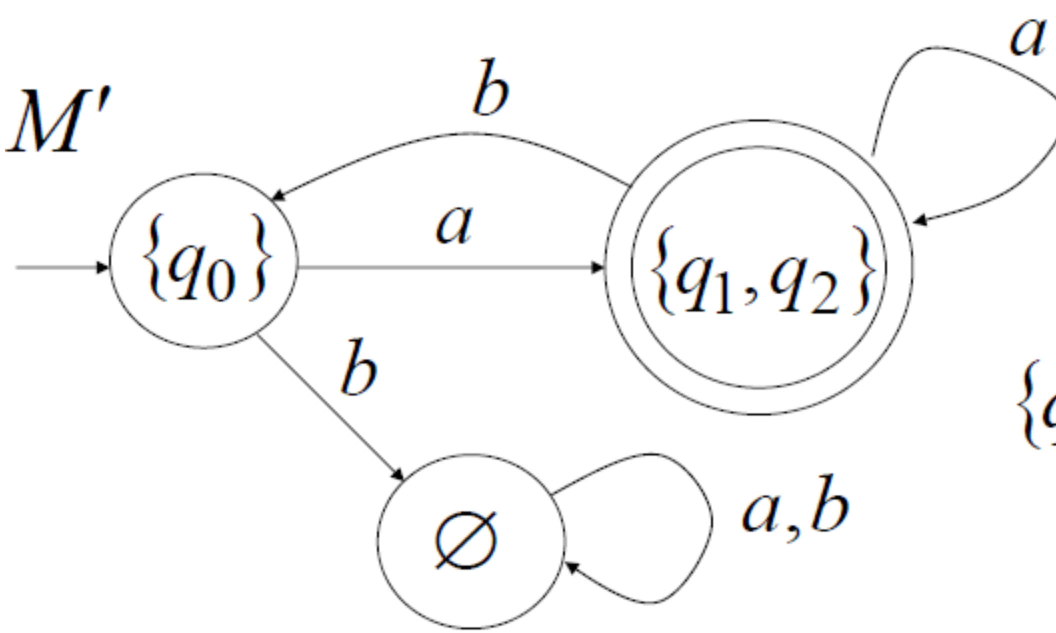
Example

- NFA



$q_1 \in F$

- DFA



$\{q_1, q_2\} \in F'$

Theorem

Take NFA M, Apply procedure to obtain DFA M'
Then M and M' are equivalent:

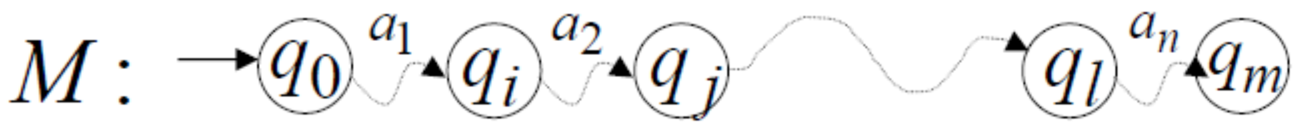
$$L(M)=L(M')$$

- To prove $L(M)=L(M')$
  We need to prove $L(M) \subseteq L(M')$ and $L(M') \subseteq L(M)$

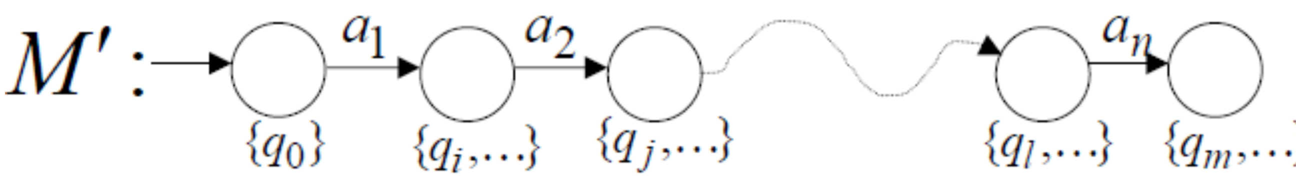
- Any string accepted by M is accepted by M'.
  - For any $w \in L(M)$, $w \in L(M')$
- Any string accepted by M' is accepted by M.
  - For any $w \in L(M')$, $w \in L(M)$

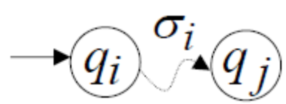
w ∈L(M) -> w ∈L(M’)

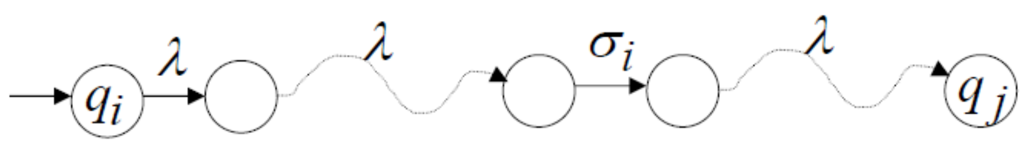
We will show that if in M: Arbitrary string $v=a_1a_2\ldots a_n$



Then



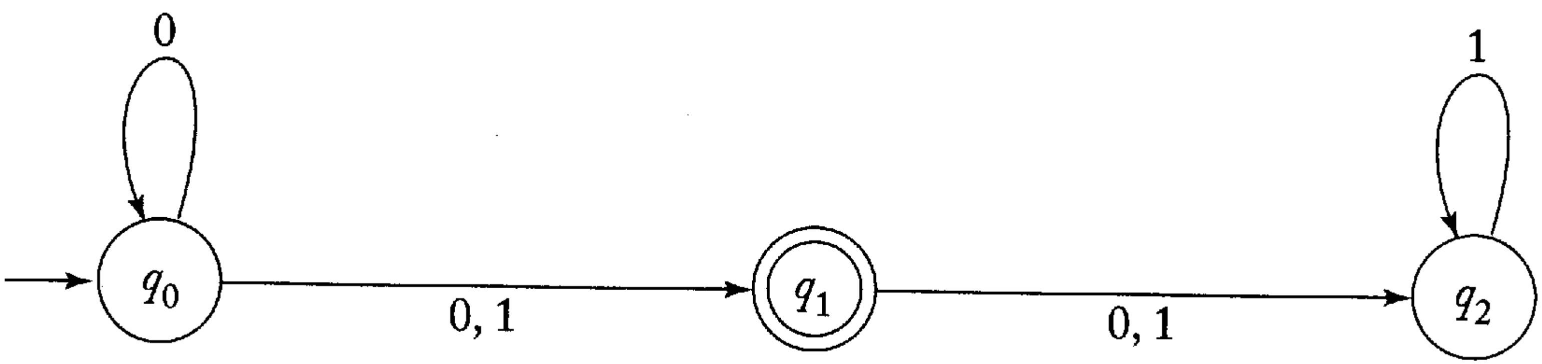
where in NFA  denotes



- Proof by induction on |v|, similar proof for w ∈L(M’) -> w ∈L(M)
- The proof is mostly based on the way we construct M’

NFA → DFA

Exercise: Convert this NFA to a DFA.



Minimal DFA’s

- We say M is minimal if there is no other DFA with a smaller number of states which also accepts L(M).

Theorem 2.4

Given any DFA M, application of the procedure Reduce (in the textbook) yields another DFA M’ such that

L(M) = L(M’)

Furthermore, M’ is minimal in the sense that there is no other DFA with a smaller number of states which also accepts L(M).