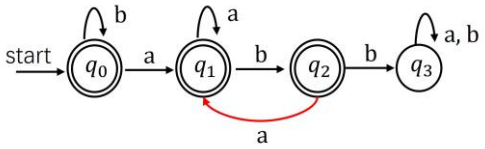
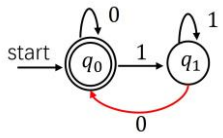
Week 2 Summary Sheet
Wanrong Yang
04/10/2024

More Examples in DFA Application

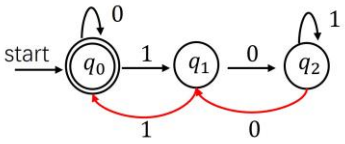
1. Not substring problem
   construct DFA for the language over $\Sigma = \{a, b\}$ that dose not contain substring abb
   - start wiht a DFA that do contain substring abb
   - identify the final accepted states



2. Binary number is even
   - design a DFA which checks whether the given binary number is even
   - binary number over $\Sigma = \{0,1\}$, eg. 0100=4(even number), 0101=5 (odd number)
   - binary strings ending with 0 means even number, and ending with 1 means odd



3. Divisible by 3
   - construct a DFA which accepts set of all binary strings divisible by 3
   - reminders are always 0, 1, 2, so 3 states q_0, q_1, q_2 in total



4. Even 0's and even 1's
   - construct a DFA to accept the language over $\Sigma = \{0,1\}$, and sepecifically in this $L = \{\omega | \omega \; has \; even \; number \; of \; 0's \; and \; even \; number \; of \; 1's\}$
   - q_0 means even 0's & even 1's
   - q_1 means even 0's & odd 1's
   - q_2 means odd 0's & even 1's
   - q_3 means odd 0's & odd 1's
5. Extended transition function for DFA
   - $\hat{\delta}(q, \omega) = p$ read as delta-hat, and ω is a string
   - after reading a string ω as an input, it will transfered from state q to state p
   - language of a DFA

     DFA, $A = (Q, \Sigma, \delta, q_0, F)$ then $L(A) = \{\omega | \hat{\delta}(q_0, \omega) \in F\}$

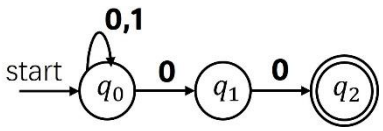
Non-Deterministic Finite Automata (NFA)

1. Introduction
   - NFA, $N = (Q, \Sigma, \delta, q_0, F)$

- keep it in mind that $\delta(q,a) = \{q_1, q, q_3, \ldots, q_k\}$
- it may have multiple next states, given a fixed current state and input

1.1 example 1, common problems

construct NFA to accept the strings of 0's and 1's that finally end with 00



try to construct a DFA to accept the strings

find at least one path leading to an accepted state, if not, leading to a rejection

empty set or state $\emptyset$ means a rejection

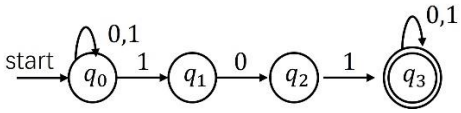
1.2 example 2, substring problems

construct NFA and DFA which accept set of all strings over an $\Sigma = \{0,1\}$, where each string contains substring 101

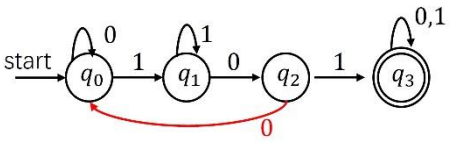
eg. $\{001011,101,11011,\ldots\ldots\}$ can be accepted

eg. $\{00100,1100,00110,\ldots\ldots\}$ can not be accepted

NFA should looks like this

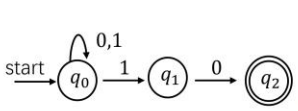


DFA should looks like this



NFA to DFA Conversion (Equivalence of NFA and DFA)

1. Introduction
   - every DFA is an NFA
   - not every NFA can be transferred into a DFA
2. Subset construction method
   - 2.1 get transition table of a given NFA
   - 2.2 calculate transition table for a DFA
     - if NFA has n states, then targeted DFA will have 2^n states in total
     - Identify starting state and final states, means just to find any state set including at least 1 original accepted state)
     - union operation
   - 2.3 identify all accessible states from starting state
   - 2.4 get the transition diagram



NFA

states	0	1
→ q_0	$\{q_0\}$	$\{q_0, q_1\}$
q_1	$\{q_2\}$	$\emptyset$
* q_2	$\emptyset$	$\emptyset$

DFA

	states	0	1
	$\emptyset$	$\emptyset$	$\emptyset$
✓	→ $\{q_0\}$	$\{q_0\}$	$\{q_0, q_1\}$
	$\{q_1\}$	$\{q_2\}$	$\emptyset$
	* $\{q_2\}$	$\emptyset$	$\emptyset$
✓	$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
	* $\{q_0, q_2\}$	$\{q_0, q_1\}$	$\{q_0\}$
	* $\{q_1, q_2\}$	$\{q_2\}$	$\emptyset$
✓	* $\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$

✓ means accessible states