CS/ECE 374 Spring 2025 Heng An Cheng (hacheng2@illinois.edu)
Homework 2 Problem 1

Solution:

(a) Regular Expression for all strings that end in 010 and contain 000 as a substring. We can break down all strings in the language into two cases:

Case 1: Strings do not end with 00010, but contain 000 as a substring and end with 010. We can use the following regular expression to represent the strings that end with 00010:

$$(0+1)^*000(0+1)^*010$$

Case 2: Strings end with 00010, which fulfill the condition of ending in 010 and containing 000 as a substring:

$$(0+1)^*00010$$

Therefore, the regular expression for all strings that end in 010 and contain 000 as a substring is:

$$(0+1)^*000(0+1)^*010 + (0+1)^*00010$$

Verification:

The first form ensures that all strings contain 000 as a substring and end with 010. However, this excludes the condition where the substring 000 combines with 010, forming 00010 at the end of the string, which corresponds to Case 2.

Therefore, using union to combine the strings from both cases will correctly match the language.

(b) Regular Expression for all strings that do not contain the subsequence 010.

We observe the following properties:

- The string can only contain a run of consecutive 0s.
- There can be any number of consecutive 1s before or after consecutive 0s.

Therefore, the regular expression for all strings that do not contain the subsequence 010 is:

$$1^*0^*1^*$$

Verification:

Since there are no occurrences of a '0' appearing after a '0' and a '1', the subsequence 010 can never occur in any string generated by this expression.

(c) Regular Expression of the complement of the language {0,01,10,101,011,111}.

There will be 5 cases. The length of the strings are 0,1,2,3 and at least 4.

- len=0 => ϵ
- len=1 => 1
- len=2 => $00, 11$
- len=3 => $000, 001, 010, 100, 110$
- len>=4 =>$(0+1)(0+1)(0+1)(0+1)^+$

So the regular expression of the complement of the language is

$$\epsilon + 1 + 00 + 11 + 000 + 001 + 010 + 100 + 110 + (0+1)(0+1)(0+1)(0+1)^+$$

(d) Discuss all possible combination of a,b,c. Define x, y, z are integer >=0.

- Case 1: $a \equiv (b+c) \equiv 0 \pmod 2$
  Let a=2x, b=2y, c=2z, which will be $(00)^*(11)^*(22)^*$
  Let a=2x, b=2y+1, c=2z+1, which will be $(00)^*1(11)^*2(22)^*$
- Case 2: $a \equiv (b+c) \equiv 1 \pmod 2$
  Let a=2x+1, b=2y+1, c=2z, which will be $0(00)^*1(11)^*(22)^*$
  Let a=2x+1, b=2y, c=2z+1, which will be $0(00)^*(11)^*2(22)^*$

So the regular expression for the language will be

$$(00)^*(11)^*(22)^* + (00)^*1(11)^*2(22)^* + 0(00)^*1(11)^*(22)^* + 0(00)^*(11)^*2(22)^*$$

(e)

- Case 1: For each of the base cases where r is one of $\emptyset, \epsilon, 0, 1$, we determine r', which represents a regular expression for $L(r) - \{\epsilon\}$.
  We can find r' as follows:

$$r' = \begin{cases} \emptyset, & \text{if } r = \emptyset \text{ or } r = \epsilon \\ 0, & \text{if } r = 0 \\ 1, & \text{if } r = 1 \end{cases}$$

- Case 2: Given $r = r_1 + r_2$ and r_1' , r_2' as regular expressions for $L(r_1) - \{\epsilon\}$ and $L(r_2) - \{\epsilon\}$ respectively.
  We can find r' as follows:

$$r' = r_1' + r_2'$$

- Case 3: Given concatenation $r = r_1 r_2$.
  We can find r' as follows:

$$r' = r_1' + r_2' + r_1' r_2'$$

- Case 4: Given $r = r_1^*$.
  We can find r' as follows:

$$r' = (r_1')^+$$

Algorithm 1 Remove ϵ from Regular Expression of L(r)

function REMOVE(r)
  if $r = \emptyset$ or $r = \epsilon$ **then**
    return $\emptyset$
  else if $r = 0$ **then**
    return 0
  else if $r = 1$ **then**
    return 1
  else if $r = r_1 + r_2$ **then**
    $r' \leftarrow \text{REMOVE}(r_1) + \text{REMOVE}(r_2)$
  else if $r = r_1 r_2$ **then**
    $r' \leftarrow \text{REMOVE}(r_1) + \text{REMOVE}(r_2) + \text{REMOVE}(r_1) \cdot \text{REMOVE}(r_2)$
  else if $r = r_1^*$ **then**
    $r' \leftarrow (\text{REMOVE}(r_1))^+$
  end if
  return r'
end function

■

CS/ECE 374 Spring 2025 — Heng An Cheng (hacheng2@illinois.edu)

Homework 2 Problem 2

Solution:

(a) State Explanation:

e = is the start state, which means the string doesn't end with 1 or 11. 1 = means the string ends with 1. 11 = means the string ends with 11, which is the accepting state.

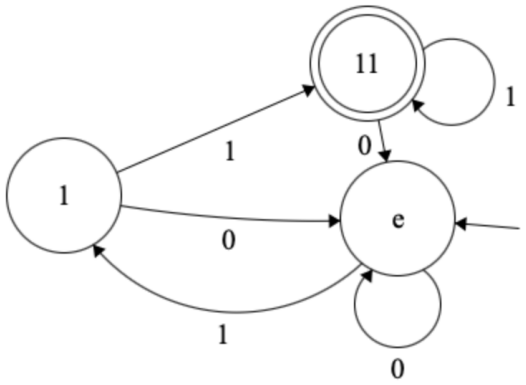

Figure 1. DFA for (a)

(b) State Explanation:

(Define n as an integer where $n \geq 0$.)

e = empty string, which is the start state

o = string "o"

o1 = string "o1"

(len $= 4n + 3$): The length of the string can be represented by $4n + 3$.

(len $= 4n$): The length of the string can be represented by $4n$.

(len $= 4n + 1$): The length of the string can be represented by $4n + 1$.

(len $= 4n + 2$): The length of the string can be represented by $4n + 2$.

Invalid: If the string starts with 1 or oo, it is definitely not in the language.

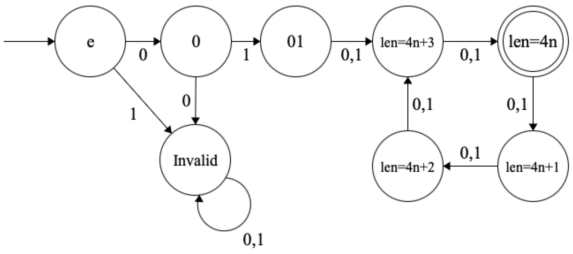

Figure 2. DFA for (b)

(c) State Explanation:

e = empty string, which is the start state and also the accepting state .

1 = substring "1", the accepting state

10 = substring "10", the accepting state

100 = substring "100", the accepting state

1001 = substring "1001", the accepting state

10011 = substring "10011", the accepting state

100110 = substring "100110", containing 100110, which is definitely not in the language.

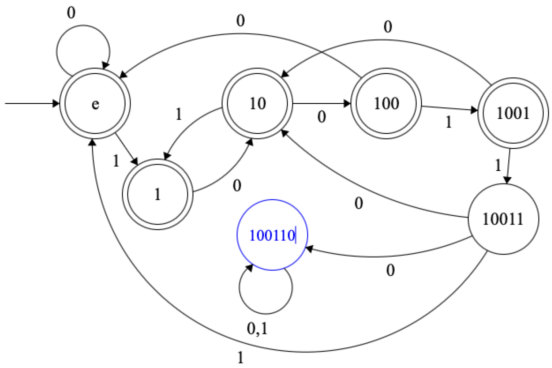

Figure 3. DFA for (c)

(d) DFA is defined as follows:

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

where:

- $Q = \{e, 1, 11, 111, 1111, 11111, 111111, 1111111, 11111111\} = \{q_0, q_1, \ldots, q_8\}$
- Start state: $q_0 = e$
- Accepting state: $F = \{q_8\}$
- Alphabet: $\Sigma = \{0, 1\}$
- Transition function δ is defined as:

$$\delta(q_i, 0) = q_0, \quad \forall i$$

$$\delta(q_i, 1) = q_{i+1}, \quad \text{for } i < 8$$

$$\delta(q_8, 1) = q_8$$

Each state represents the number of consecutive $1s$ at the end of the input string.

(e) DFA is defined as follows:

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

where:

- $Q = \{(a, b, c) \mid 0 \leq a < 2, 0 \leq b < 7, 0 \leq c < 4\}$
  (a, b, c) represents:
  - $a = \#(0, w) \mod 3$
  - $b = \#(1, w) \mod 7$
  - $c = |w| \mod 4$
- Start state: $q_0 = (0, 0, 0)$
- Accepting state: $F = \{(a, b, c) \mid (a + b) \equiv c \pmod 4\}$
- Alphabet: $\Sigma = \{0, 1\}$
- Transition function δ is defined as:

$$\delta((a, b, c), 0) = (((a + 1) \mod 3), b, ((c + 1) \mod 4))$$

$$\delta((a, b, c), 1) = (a, ((b + 1) \mod 7), ((c + 1) \mod 4))$$

  Each state record the number of $0s$ modulo 3, number of $1s$ modulo 7 and the number of length of w modulo 4.

■