Regular Expressions and Operations

• 2017/08/31

• Chapter 1.2 & 1.3 in Sipser

➢ Announcement:

❑ Many thanks to students who have responded so far! There is still time to respond to the poll. (Today’s slides are adjusted accordingly.)

❑ Piazza registration: http://piazza.com/asu/fall2017/cse355

❑ Slides for this lecture are here:

http://www.public.asu.edu/~yzhan442/teaching/CSE355/Lectures/REO.pdf

❑ HW2 on NFA will be released soon; HW1 due tonight 11:59PM

◆ Happy Labor Day!
Last time

• Nondeterministic finite automata
  (introducing non-determinism into FA)
  o Definition of NFA
  o Computation of NFA
  o Equivalence of DFA and NFA
    ✗ A language is regular if and only if some NFA recognizes it
Previously

- Finite automata
  - Definition of FA
  - Computation of FA
  - Regular language
    - Regular language and FA
    - Regular operations
  - Design an FA

- Nondeterministic finite automata
  (introducing non-determinism into FA)
  - Definition of NFA
  - Computation of NFA
  - Equivalence of DFA and NFA
Previously

• Finite automata
  - Definition of FA
  - Computation of FA
  - Regular language
    - Regular language and FA
    - Regular operations
  - Design an FA

• Nondeterministic finite automata
  (introducing non-determinism into FA)
  - Definition of NFA
  - Computation of NFA
  - Equivalence of DFA and NFA
Previously

- Finite automata
  - Definition of FA
  - Computation of FA
  - Regular language
    - Regular language and FA
    - Regular operations
  - Design an FA

- Nondeterministic finite automata
  (introducing non-determinism into FA)
  - Definition of NFA
  - Computation of NFA
  - Equivalence of DFA and NFA
Outline for today

• Regular expressions (REs)
  o Regular operations and FA
  o RE examples
  o Equivalence of FA and REs (next class)

➢ Goals:
  o Learn how to write formal proofs
  o Understand more about regular operations
  o Learn to use REs
Definition of Regular Expression

**Definition 1.52**

Say that \( R \) is a *regular expression* if \( R \) is

1. \( a \) for some \( a \) in the alphabet \( \Sigma \),
2. \( \varepsilon \),
3. \( \emptyset \),
4. \( (R_1 \cup R_2) \), where \( R_1 \) and \( R_2 \) are regular expressions,
5. \( (R_1 \circ R_2) \), where \( R_1 \) and \( R_2 \) are regular expressions, or
6. \( (R_1^*) \), where \( R_1 \) is a regular expression.

In items 1 and 2, the regular expressions \( a \) and \( \varepsilon \) represent the languages \( \{a\} \) and \( \{\varepsilon\} \), respectively. In item 3, the regular expression \( \emptyset \) represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages \( R_1 \) and \( R_2 \), or the star of the language \( R_1 \), respectively.
Definition of Regular Expression

**Definition 1.52**

Say that $R$ is a *regular expression* if $R$ is

1. $a$ for some $a$ in the alphabet $\Sigma$,
2. $\varepsilon$,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where $R_1$ and $R_2$ are regular expressions,
5. $(R_1 \circ R_2)$, where $R_1$ and $R_2$ are regular expressions, or
6. $(R_1^*)$, where $R_1$ is a regular expression.

In items 1 and 2, the regular expressions $a$ and $\varepsilon$ represent the languages $\{a\}$ and $\{\varepsilon\}$, respectively. In item 3, the regular expression $\emptyset$ represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages $R_1$ and $R_2$, or the star of the language $R_1$, respectively.

In order to prove equivalence, we must first convince ourselves that RL is closed under regular operations!

Definition - A collection (or set) is **closed** under an operation if applying this operation to members of the collection returns a member in the collection.
Regular operations

**Definition 1.23**

Let $A$ and $B$ be languages. We define the regular operations *union*, *concatenation*, and *star* as follows:

- **Union**: $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$.
- **Concatenation**: $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$.
- **Star**: $A^* = \{x_1x_2\ldots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$.

**Example 1.24**

Let the alphabet $\Sigma$ be the standard 26 letters \{a, b, \ldots, z\}. If $A = \{\text{good, bad}\}$ and $B = \{\text{boy, girl}\}$, then

- $A \cup B = \{\text{good, bad, boy, girl}\}$,
- $A \circ B = \{\text{goodboy, goodgirl, badboy, badgirl}\}$, and
- $A^* = \{\epsilon, \text{good, bad, goodgood, goodbad, badgood, badbad, goodgoodgood, goodgoodbad, goodbadgood, goodbadbad, \ldots}\}$. 

\[\]
Closed under union

**Theorem 1.25**

The class of regular languages is closed under the union operation.

1. Given any RLs $A_1$ and $A_2$
2. Based on the definition, we can construct DFA $M_1$ for $A_1$ and $M_2$ for $A_2$
3. Prove that $A_1 \cup A_2$ is also a RL
4. Construct an NFA (DFA) $M$ to simulate both $M_1$ and $M_2$ at the same time and accept if either one accepts
   - Keep a copy of both $M_1$ and $M_2$; for every step in $M$, run a step from $M_1$ and then a step in $M_2$; $M$ accepts if either $M_1$ or $M_2$ accepts; otherwise, reject

   ➤ This works now with NFA!
Closed under union

**Theorem 1.25**

The class of regular languages is closed under the union operation.

1. Given any RLs $A_1$ and $A_2$
2. Based on the definition, we can construct DFA $M_1$ for $A_1$ and $M_2$ for $A_2$

1. Prove that $A_1 \cup A_2$ is also a RL
2. Construct an NFA (DFA) $M$ to simulate both $M_1$ and $M_2$ at the same time and accept if either one accepts
   - Keep a copy of both $M_1$ and $M_2$; for every step in $M$, run a step from $M_1$ and then a step in $M_2$; $M$ accepts if either $M_1$ or $M_2$ accepts; otherwise, reject

- **This works now with NFA!**

Construct $M = (Q, \Sigma, \delta, t_0, F)$ to recognize $L(M_1) \cup L(M_2)$

$Q = \{t_0\} \cup Q_1 \cup Q_2$

$\Sigma$ is unchanged

$\delta(q, a) =$

$q \in Q, a \in \Sigma$

$t_0$ is the start state of $M$

$F = F_1 \cup F_2$ is the set of accept states
THEOREM 1.25

The class of regular languages is closed under the union operation.

1. Given any RLs A1 and A2
2. Based on the definition, we can construct DFA M1 for A1 and M2 for A2

Prove that A1 ∪ A2 is also a RL

Construct an NFA (DFA) M to simulate both M1 and M2 at the same time and accept if either one accepts

- Keep a copy of both M1 and M2; for every step in M, run a step from M1 and then a step in M2; M accepts if either M1 or M2 accepts; otherwise, reject

This works now with NFA!

Construct \( M = (Q, \Sigma, \delta, t_0, F) \) to recognize \( L(M_1) \cup L(M_2) \):

- \( Q = \{t_0\} \cup Q_1 \cup Q_2 \)
- \( \Sigma \) is unchanged

\[
\delta(q, a) = \begin{cases} 
\{\delta_1(q, a)\} & q \in Q_1 \text{ and } a \in \Sigma \\
\{\delta_2(q, a)\} & q \in Q_2 \text{ and } a \in \Sigma \\
\{q_0, s_0\} & q = t_0 \text{ and } a = \varepsilon 
\end{cases}
\]

- \( q \in Q, a \in \Sigma^* \)
- \( t_0 \) is the start state of \( M \)
- \( F = F_1 \cup F_2 \) is the set of accept states
Closed under concatenation

**Theorem 1.47**  

The class of regular languages is closed under the concatenation operation.

1. Given any RLs $A_1$ and $A_2$
2. Based on the definition, we can construct DFA $M_1$ for $A_1$ and $M_2$ for $A_2$

1. Prove that $A_1 \circ A_2$ is also a RL
2. Construct an NFA $M$ to accept $A_1 \circ A_2$

---

Diagram showing the construction of an NFA from two DFAs.
Closed under concatenation

**Theorem 1.47**

The class of regular languages is closed under the concatenation operation.

1. Given any RLs $A_1$ and $A_2$
2. Based on the definition, we can construct DFA $M_1$ for $A_1$ and $M_2$ for $A_2$
   
   1. Prove that $A_1 \circ A_2$ is also a RL
   2. Construct an NFA $M$ to accept $A_1 \circ A_2$

![Diagram of DFAs and NFAs](image-url)
Closed under concatenation

**Theorem 1.47**

The class of regular languages is closed under the concatenation operation.

1. Given any RLs A1 and A2
2. Based on the definition, we can construct DFA M1 for A1 and M2 for A2

   \[ M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1) \]
   \[ M_2 = (Q_2, \Sigma, \delta_2, s_0, F_2) \]

   Construct \( M = (Q, \Sigma, \delta, t_0, F) \) to recognize \( L(M_1) \cup L(M_2) \)
   \[ Q = Q_1 \cup Q_2 \]
   \( \Sigma \) is unchanged
   \[ t_0 = q_0 \]
   \[ \delta(q, a) = \]
   \[ q \in Q, a \in \Sigma_\varepsilon \]
   \[ F = F_2 \]
Closed under concatenation

The class of regular languages is closed under the concatenation operation.

1. Given any RLs A1 and A2
2. Based on the definition, we can construct DFA M1 for A1 and M2 for A2

\[ M_1 = (Q_1, \Sigma, \delta_1, q_0, F_1) \]
\[ M_2 = (Q_2, \Sigma, \delta_2, s_0, F_2) \]

Construct \( M = (Q, \Sigma, \delta, t_0, F) \) to recognize \( L(M_1) \cup L(M_2) \)
\[ Q = Q_1 \cup Q_2 \]
\( \Sigma \) is unchanged
\[ t_0 = q_0 \]
\[ \delta(q, a) = \begin{cases} 
\{\delta_1(q, a)\} & q \in Q_1 \text{ and } a \in \Sigma \\
\{s_0\} & q \in F_1 \text{ and } a = \varepsilon \\
\{\delta_2(q, a)\} & q \in Q_2 \text{ and } a \in \Sigma
\end{cases} \]
\[ F = F_2 \]
The class of regular languages is closed under the star operation.

1. Given any RL $A$
2. Based on the definition, we can construct a DFA $M$ for $A$

1. Prove that $A^*$ is also a RL
2. Construct an NFA $M$ to accept $A^*$
Closed under star

**Theorem 1.49**

The class of regular languages is closed under the star operation.

1. Given any RL $A$
2. Based on the definition, we can construct a DFA $M$ for $A$

1. Prove that $A^*$ is also a RL
2. Construct an NFA $M$ to accept $A^*$
**Theorem 1.49**

The class of regular languages is closed under the star operation.

1. Given any RL $A$, construct a DFA $M$ for $A$.
2. Based on the definition, we can construct a DFA $M$ for $A$.

Prove that $A^*$ is also a RL.

Construct an NFA $M$ to accept $A^*$.

1. Prove that $A^*$ is also a RL.
2. Construct an NFA $M$ to accept $A^*$.

Construct $M = (Q, \Sigma, \delta, t_0, F)$ to recognize $A^*$.

$Q = \{t_0\} \cup Q_1$

$\Sigma$ is unchanged.

$\delta(q, a) =$

$q \in Q, a \in \Sigma$ (start state)

$t_0$ is the start state

$F = \{t_0\} \cup F_1$
Closed under star

**Theorem 1.49**

The class of regular languages is closed under the star operation.

1. Given any RL $A$
2. Based on the definition, we can construct a DFA $M$ for $A$

1. Prove that $A^*$ is also a RL
2. Construct an NFA $M$ to accept $A^*$

Construct $M = (Q, \Sigma, \delta, t_0, F)$ to recognize $A^*$

$Q = \{ t_0 \} \cup Q_1$

$\Sigma$ is unchanged

\[
\delta(q, a) = \begin{cases} 
\{ \delta_1(q, a) \} & q \in Q_1 \text{ and } a \in \Sigma \\
\{ q_0 \} & q \in F_1 \text{ and } a = \varepsilon \\
\{ q_0 \} & q = t_0 \text{ and } a = \varepsilon 
\end{cases}
\]

$q \in Q, a \in \Sigma_\varepsilon$

$t_0$ is the start state

$F = \{ t_0 \} \cup F_1$
Outline for today

• Regular expressions (REs)
  o Regular operations and FA
  o RE examples
  o Equivalence of FA and REs (next class)

➤ Goals:
  o Learn how to write formal proofs
  o Understand more about regular operations
  o Learn to use REs
Definition of Regular Expression

**Definition 1.52**

Say that $R$ is a **regular expression** if $R$ is

1. $a$ for some $a$ in the alphabet $\Sigma$,
2. $\epsilon$,
3. $\emptyset$,
4. $(R_1 \cup R_2)$, where $R_1$ and $R_2$ are regular expressions,
5. $(R_1 \circ R_2)$, where $R_1$ and $R_2$ are regular expressions, or
6. $(R_1^*)$, where $R_1$ is a regular expression.

In items 1 and 2, the regular expressions $a$ and $\epsilon$ represent the languages $\{a\}$ and $\{\epsilon\}$, respectively. In item 3, the regular expression $\emptyset$ represents the empty language. In items 4, 5, and 6, the expressions represent the languages obtained by taking the union or concatenation of the languages $R_1$ and $R_2$, or the star of the language $R_1$, respectively.
Regular Expression

• Notes about RE
  ➢ A regular expression represents a language
    ☐ When we use RE $R$ with regular operations, it should be interpreted as $L(R)$

  ➢ $+$ and $*$ $> \circ > \cup$
    ☐ $0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 = \{w \mid w$ starts and ends with the same symbol$\}$

  ➢ In practice: $(+ \cup - \cup \varepsilon)(D^+ \cup D^+.D^* \cup D^*.D^*)$
Regular Expression

- More notes
  - \( R^+ = RR^* \)
  - \( R^* = R^+ \cup \{\varepsilon\} \)
  - \( \emptyset^* = \{\varepsilon\} \)
  - \( R \circ \varepsilon = R \)
  - \( R \circ \emptyset = \emptyset \)
  - \( R \cup \varepsilon = R? \)
  - \( R \circ \emptyset \) is never equal to \( R? \)
  - \( R_1(R_2 \cup R_3) = R_1R_2 \cup R_1R_3 \)

What about \( R_1(R_2 \cup R_3)^* \)
RE example

- Example: $L(M) = \{ w \mid w \in \{0, 1\}^* \text{ and } 001 \text{ is a suffix of } w \}$
**RE example**

- Example: \( L(M) = \{ w \mid w \in \{0, 1\}^* \text{ and } 001 \text{ is a suffix of } w \} \)

\[
(0 \cup 1)^*001 \quad \text{or equivalently } \{0, 1\}^*001
\]
RE example
RE example

$(aa \cup aaaa)^*b$ or equivalently $\{aa, aaaa\}^*b$
RE example
RE example

\[ a^* \cup (ba^* (a \cup b)a^*)^* \]
Summary

• Regular expression are defined to be closed under regular operations

• We have shown in this lecture that
  o Regular languages are also closely under regular operations
  o We can express FA using REs
Outline for today

• Regular expressions (REs)
  o Regular operations and FA
  o RE examples
  o Equivalence of FA and REs (next class)

Goals:
  o Learn how to write formal proofs
  o Understand more about regular operations
  o Learn to use REs

• Reading assignment for the next class:
  o Sipser Sec. 1.3 (page 66-77 Equivalence with FA) – Quiz link
    will be sent out; due date is before the beginning of the next class