CSCI 1010
Models of Computation

Lecture 14
Regular Expressions
Overview

• Review of definition of regular expressions and regular languages.
• Proof that languages defined by r.e.’s are recognized by NFSMs (equally FSMs).
Operations on Sets

• Regular expressions use **union**, **concatenation**, and **Kleene closure** on sets. Let \( L_1, L_2 \subseteq \Sigma^* \).

• Concatenation: \( L_1 \cdot L_2 = \{ uv \mid u \in L_1, v \in L_2 \} \).
  
  – If \( L_1 = \{10,0\} \), \( L_2 = \{aa, b\} \), \( L_1 \cdot L_2 = \{10aa, 0aa, 10b, 0b\} \)

• Powers of a set: \( L^2 = L \cdot L \), \( L^3 = L \cdot L \cdot L \), \( L^4 = L \cdot L^3 \), ...

• Kleene closure: \( L^* = L^0 \cup L \cup L^2 \cup L^3 \cup \ldots \) where \( L^0 = \{\varepsilon\} \) where \( \varepsilon \) is the empty string.
  
  – \( L_1^* = \{\varepsilon, 10,0,1010,100,010,00,\ldots\} \)

• Positive closure: \( L^+ = L \cup L^2 \cup L^3 \cup \ldots \)
Regular Expressions

• A regular expression (r.e.) and the associated languages are defined below.
  – $\emptyset$ is the r.e. denoting the empty set.
  – $\varepsilon$ (bold) denotes set $\{\varepsilon\}$ containing empty letter
  – $a$ denotes the set $\{a\}$ for $a$ in $\Sigma$
  – If $r$ and $s$ are r.e.’s denoting sets $R$ and $S$, then $rs$, $r+s$, $r^*$ are also r.e.’s denoting the sets $R\cdot S$, $R \cup S$, and $R^*$, respectively.

• E.g. The following are r.e.’s over $\{a,b,c\}$:
  – $ac$, $ab^*$, $(ab+b)^*$, $a^* + ab^*a$
Regular Languages and R.E.’s

• The languages denoted by r.e.’s are called regular languages.
  – \{0+1\}^* denotes the set of all strings over \{0,1\} which was previously denoted \{0,1\}^*.
  – What is the language \((0)(1^*)(0)+(1)\)?

• We now drop boldface on letters and assume that the meaning is understood.
Properties of Regular Expressions

1. $r \emptyset = \emptyset r = \emptyset$
2. $r \varepsilon = \varepsilon r = r$
3. $r + \emptyset = \emptyset + r = r$
4. $r + r = r$
5. $r + s = s + r$
6. $r (s + t) = rs + rt$
7. $(r + s)t = rt + st$
8. $r (s t) = (rs) t$
9. $\emptyset^* = \varepsilon$
10. $\varepsilon^* = \varepsilon$
11. $(\varepsilon + r)^* = r^*$
12. $(\varepsilon + r)^* = r^*$
13. $r^*(\varepsilon + r) = (\varepsilon + r)r^* = r^*$
14. $r^*s + s = r^*s$
15. $r (s r)^* = (rs)^*r$
16. $(r + s)^* = (r^*s)^*r^* = (s^*r)^*s^*$
Nesting Depth of a Regular Expression

• Consider r.e.s for *concatenation* $(rs)$, *union* $(r+s)$, and *Kleene closure* $(r^*)$.

• **Definition** The nesting depth $d(r)$ of an r.e. $r$ is defined recursively as follows:
  
  - $d(r) = 0$ for $r = \varepsilon, \emptyset$, or $a$ for some letter $a$.
  - $d(rs) = 1 + \max(d(r), d(s))$
  - $d(r+s) = 1 + \max(d(r), d(s))$
  - $d(r^*) = 1 + d(r)$
Regular Expressions and FSMs

**Theorem** Given an r.e. \( r \), there is an FSM recognizing the language denoted by \( r \).

**Proof** By *induction* on the nesting depth of \( r \).

**Base case:**
Depth is 0 for \( r = \varepsilon, \emptyset, \) or \( a \) for some letter \( a \). Each r.e. recognized by an NFSM:
Inductive Hypothesis

r.e.’s $rs$, $r+s$, $r^*$ have depth one more than the maximum depth of $r$ & $s$ or $r$ alone, respectively. Assume that there exist NFSMs for $r$ and $s$. We construct NFSMs for $rs$, $r+s$, and $r^*$. 
Concatenation *rs*

- **NFSM M for rs** uses NFSMs $M_1 = (\Sigma, Q_1, \delta_1, s_1, F_1)$ for $r$ and $M_2 = (\Sigma, Q_2, \delta_2, s_2, F_2)$ for $s$.
- $s_1$ is *M*’s start state. $F_2$ are accept states of *M*.
- Insert an **ε-transition** from each final state of $M_1$ to the start state $s_2$ of $M_2$.
  - An ε-transition occurs on no input.
Concatenation \( rs \)

- Strings \( uv \) are in \( rs \) iff \( u \) takes \( M_1 \) to one of its accept states and \( v \) takes \( M_2 \) from its initial state \( s_2 \) to a final state of \( M_2 \). This is achieved by inserting an \( \varepsilon \)-transition between a final state of \( M_1 \) and \( s_2 \).
Concatenation

• Theorem If NFSM $M_1$ recognizes $L_1$ and NFSM $M_2$ recognizes $L_2$, then there is an NFSM $M_3$ that recognizes $L_1 \cdot L_2$. 
Removing $\varepsilon$-Transitions

• To remove $\varepsilon$-transitions, add an edge from accept state $f_j$ of $M_1$ to a state $q_j$ of $M_2$ with label $x$ if there is such an edge from $s_2$ to $q_j$ of $M_2$.
• It follows that the new machine accepts only strings $uv$ where $u$ is in $r$ and $v$ is in $s$. 
Union Operation $r+s$

• Goal: Construct an **NFSM M for $r+s$** from the NFSMs $M_1$ and $M_2$ for $r$ and $s$.
• Let $s_0 = \text{new start state}$ for $M$. Insert $\varepsilon$-transition from $s_0$ to both start states $s_1$ of $M_1$ and $s_2$ of $M_2$.
• The **accept states of M** are $F = F_1 \cup F_2$.
• $M$ can non-deterministically choose to start in $M_1$ or $M_2$. Thus, it recognizes $r+s$. 
Union Operation $r+s$

- To remove $\varepsilon$-transitions, for $j = 1, 2$ add an edge from $s_0$ to state $q_t$ of $M_j$ with label $x$ if there is such an edge from $s_j$ to $q_t$ in $M_j$.
- The first input causes a branch to either $M_1$ or $M_2$. Thus, $M$ accepts only strings in $r+s$. 
Union

• **Theorem** If NFSM $M_1$ recognizes $L_1$ and NFSM $M_2$ recognizes $L_2$, then there is an NFSM $M_3$ that recognizes $L_1 \cup L_2$. 
Kleene Closure $r^*$

- **NFSM $M$ for $r^*$** built from NFSMs $M_1$ for $r$.
- $M$ has **new start state** $s_0$. $M_1$ has start state $s_1$.
- Insert $\varepsilon$-transition from $s_0$ to start state $s_1$.
- Insert $\varepsilon$-transitions from each final state of $M_1$ to the start state $s_1$ of $M_1$.
- **$M$ has accept states** $\{s_0\} \cup F_1$.
- Thus, $M$ accepts $\varepsilon$, all strings in $r$, $r^2$, $r^3$, ...
Kleene Closure $r^*$

- To remove $\epsilon$-transitions, add edge from $s_0$ and $f \in F_1$ to state $q_t$ of $M_1$ with label $x$ if there is such an edge from $s_1$ to $q_t$ in $M_1$. 
Kleen Closure

• **Theorem** If NFSM $M_1$ recognizes $L_1$, then there is an NFSM $M_2$ that recognizes $L_1^*$. 
Application of NFSM Constructions

• Construct $M$ to recognize $r = 10^*+0$.
  – Decompose into $r = r_1r_2 + r_3$, $r_1 = 1$, $r_2 = r_4^*$, $r_3 = 0$, $r_4 = 0$.
  – Construct NFSMs to recognize
    • 0
    • 1
    • 0*
    • 10*
    • 10^*+0
NFSMs Recognizing 0, 1, 0*
NFSMs Recognizing 10* and 10* + 0

• What transitions are missing in this diagram?
DFSM Recognizing $10^*+0$
Exercise

• Let $\Sigma = \{0,1,2\}$ and let $L$ be the language over $\Sigma$ that contains each string $w$ of length 1 or more that ends with a symbol not found elsewhere in $w$.

• Construct an NFSM that accepts $L$. 
Constructing an NSFM for L

• An r.e. for $L = (\Sigma-1)^*1 + (\Sigma-2)^*2 + (\Sigma-3)^*3$

• This machine is clearly non-deterministic.
Constructing a DFSM for L
Morale of this Story

• If you can define a regular language with r.e.’s, you can construct an FSM to recognize it.
• This FSM can be used to reject strings that do not conform to the definition.
Review

• Review of definition of regular expressions and regular languages.
• Proof that languages defined by r.e.’s are recognized by NFSMs (equally FSMs).