Notes for Lecture 3

Regular Expressions – RE

- Reading: Chapter 1 done/Chapter 2 scan/Chapter 3 start – all in Dragon book
- Some Theory – don’t confuse the notation with that used in flex
- Goal is to show that REs –
  - can be recognized by DFAs
  - how flex programs can be constructed in an automatic fashion
- Strings
  - \{\} denotes a set (of symbols) / alphabet of symbols for use in making strings
  - empty string is denoted by \(\varepsilon\)
  - string/sentence/word
  - \(|s|\) length in symbols of string \(s\)
  - \(xy\) is the concatenation of strings \(x\) and \(y\) (think of multiply)
  - \(a^n = a \cdot a \cdot \ldots \cdot a\) (identity for strings)
  - \(a^n|a|^m\) ? \(m > 1\)
  - prefix/suffix/proper substring
- Languages
  - sets of strings
    - \(L \cup M = \{s | s \text{ is in } L \text{ or } M\}\)
    - \(L \cap M = \{s | s \text{ is in } L \text{ and } s \text{ is in } M\}\)
    - \(L^* = \cup_{n=0}^{\infty} L^n\) Kleene closure
    - \(L^+\) positive closure
Regular Expressions – RE

- used to define sets of strings that are type 3 languages in Chomsky hierarchy
- defined over an alphabet $\Sigma$
- $a$ is in $\Sigma$, $a$ is the RE for $\{a\}$ / also the string $a$
- $r$ and $s$ are REs denoting languages $L(r)$ and $L(s)$
  - $r|s$ - RE for language $L(r) \cup L(s)$
  - $rs$ - RE for language $L(r) \cap L(s)$
  - $r^*$ - RE for language $L(r)^*$
- Above assume precedence ordering: * concat |
  - over ride with ()

- Examples
  - $(a|b)(a|b) = \{aa, ab, ba, bb\}$
  - $a | ba | b$ ?
  - $a | a^*b$ ?
  - $(0|1)^*$

- Algebra
  - $r|s = s|r$
  - $r|(s|t) = (r|s)|t$
  - $r(st) = (rs)t$
  - $(s|t)r = sr|tr$
  - $\varepsilon r = r = r \varepsilon$
  - $r^* = (r|\varepsilon)^*$
  - $r^+ = r^*$
  - $r^* = r^*$ (definition)

Regular Expressions – RE

- Regular definitions
  - $d_1 \rightarrow r_1$
  - $d_2 \rightarrow r_2$
  - ...
  - $d_n \rightarrow r_1$
  - $r_i$ is a RE over $\Sigma \cup \{d_1, \ldots, d_n\}$
- Example – numbers
  - digit $\rightarrow 0|1|2|3|\ldots|9$
  - digits $\rightarrow$ digit digits*
  - optional_frisk $\rightarrow$ digit $|\varepsilon$
  - optional_expr $\rightarrow (E (+ | - | \varepsilon ) dig)$ $|\varepsilon$
  - num $\rightarrow$ digits optional_fisk optional_expr
Regular Expressions and Finite State Automata

- DFA that accepts $a | c a^*b$

• definition
  - states $S$, with start state, and accepting states
  - input symbols $∑$
  - transition function $\text{move(state, input symbol)} → \text{next state}$

Non-deterministic Finite State Automata – NFA

- Easier to describe REs
- Equivalent to DFAs – there is a conversion algorithm
- Key differences
  - allow $ε$-transitions
  - allow more than one transition out of a state on a particular input

- always chooses correctly if there is a path to an accepting state
- next state is a set rather than a function
Non-deterministic Finite State Automata – NFA

- Example: \( RE = aa^* \mid b \mid ab \)

- Subset construction – \( \{ \text{move}(s, a) \} \) is relabeled as a new state
- Problem – handling \( \epsilon \)-transitions
- Any state reachable by an \( \epsilon \)-transition is “part” of the state
- Leads to the idea of \( \epsilon \)-closure
- Any state reachable from \( s \) by an \( \epsilon \)-transitions is in the closure
- Example \( \epsilon \)-closure\((1) = \{1, 2, 3, 5\}\)
- Create a new state \( A = \{1, 2, 3, 5\} \) and examine transitions out of it
- \( \text{move}(A, a) = \{3, 6\} \)
- Call this a new subset state \( B = \{3, 6\} \)
- \( \text{move}(A, b) = \{4\} \)
- \( \text{move}(B, a) = \{6\} \)
- \( \text{move}(B, b) = \{4\} \)
- Complete by checking \( \text{move}(4, a); \text{move}(4, b); \text{move}(6, a); \text{move}(6, b) \)

Converting NFA → DFA

- Subset construction – \( \{ \text{move}(s, a) \} \) is relabeled as a new state
- Problem – handling \( \epsilon \)-transitions
- Any state reachable by an \( \epsilon \)-transition is “part” of the state
- Leads to the idea of \( \epsilon \)-closure
- Any state reachable from \( s \) by an \( \epsilon \)-transitions is in the closure
- Example \( \epsilon \)-closure\((1) = \{1, 2, 3, 5\}\)
- Create a new state \( A = \{1, 2, 3, 5\} \) and examine transitions out of it
- \( \text{move}(A, a) = \{3, 6\} \)
- Call this a new subset state \( B = \{3, 6\} \)
- \( \text{move}(A, b) = \{4\} \)
- \( \text{move}(B, a) = \{6\} \)
- \( \text{move}(B, b) = \{4\} \)
- Complete by checking \( \text{move}(4, a); \text{move}(4, b); \text{move}(6, a); \text{move}(6, b) \)
ε-closure(0) = (0, 1, 2, 4, 7) = A
• Now examine transitions out of A – move(A, a) and move(A, b)
• Always include the ε-closure of any state as part of that state – ε-closure(move(A, a)) etc.
• ε-closure(move(A, a)) = ε-closure((3, 8)) = {1, 2, 3, 4, 6, 7, 8} = B
• ε-closure(move(A, b)) = ε-closure((5)) = {1, 2, 4, 5, 6, 7} = D
• Now examine transitions out of B – move(B, a) and move(B, b)
• ε-closure(move(B, a)) = ε-closure({3, 8}) = {1, 2, 3, 4, 6, 7, 8} = B
• ε-closure(move(B, b)) = ε-closure({5, 9}) = {1, 2, 3, 4, 5, 6, 7, 9} = C
• ε-closure(move(C, a)) = ε-closure((3, 8)) = {1, 2, 3, 4, 6, 7, 8} = B
• ε-closure(move(C, b)) = ε-closure((5)) = {1, 2, 4, 5, 6, 7} = D
• ε-closure(move(D, a)) = ε-closure((3, 8)) = {1, 2, 3, 4, 6, 7, 8} = B
• ε-closure(move(D, b)) = ε-closure((5)) = D

ε-closure(0) = (0, 1, 2, 4, 7) = A
NDFA → DFA cont.
Converting a RE → NFA : Thompson’s Construction

• for ε

\[ i \xrightarrow{\varepsilon} f \]

• for a

\[ i \xrightarrow{a} f \]

Given REs s and t – N(s) and N(t) are NFAs

NFA(s | t)

• New start \( i \) \( \varepsilon \)-transitions to the start states of N(s) and N(t)
• \( \varepsilon \)-transitions from the final/accepting states of N(s) and N(t) to the new final state f
Converting a RE → NFA: Thompson's Construction cont.

- Given REs s and t – N(s) and N(t) are NFAs
- NFA(s t)