Announcements

- P1 part 2 due on Thursday
- H1 posted. Due on Sep 20th.
Nondeterministic Finite Automata

CS 536
Explore NFAs

Claim: NFAs add no power to DFAs

Epsilon transitions

Claim: Epsilon transitions add no power

Regular expressions
NFMs, formally

\[ (Q, \Sigma, \delta, q, F) \]

- finite set of states \( Q \)
- the alphabet (characters) \( \Sigma \)
- start state \( q \in Q \)
- final states \( F \subseteq Q \)
- transition function \( \delta : Q \times \Sigma \rightarrow 2^Q \)
NFA

To check if string is in $L(M)$ of NFA $M$, simulate set of choices it could make
NFA == DFA

Claim: \( L(NFA) = L(DFA) \)

Idea: we can only be in finitely many subsets of states at any one time

\[ 2^{\left| Q \right|} \text{ possible combinations of states} \]

Why?
Why $2^{|Q|}$ states?

**Build** DFA that tracks set of states the NFA is in!

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Define: let $\text{succ}(s, c)$ be the set of choices the NFA could make in state $s$ with character $c$

\[
\begin{align*}
\text{succ}(A, x) &= \{A, B\} \\
\text{succ}(A, y) &= \{A\} \\
\text{succ}(B, x) &= \{C\} \\
\text{succ}(B, y) &= \{C\} \\
\text{succ}(C, x) &= \{D\} \\
\text{succ}(C, y) &= \{D\}
\end{align*}
\]
Build new DFA $M'$ where $Q' = 2^Q$

To build DFA: Add an edge from state $S$ on character $c$ to state $S'$ if $S'$ represents the union of states that all states in $S$ could possibly transition to on input $c$.

- $\text{succ}(A,x) = \{A,B\}$
- $\text{succ}(A,y) = \{A\}$
- $\text{succ}(B,x) = \{C\}$
- $\text{succ}(B,y) = \{C\}$
- $\text{succ}(C,x) = \{D\}$
- $\text{succ}(C,y) = \{D\}$
ε-transitions

Eg: $x^n$, where $n$ is even or divisible by 3

Useful for taking union of two FSMs

In example, left side accepts even $n$; right side accepts $n$ divisible by 3
Eliminating $\varepsilon$-transitions

We want to construct $\varepsilon$-free FSM $M'$ that is equivalent to $M$

**Definition:**
$\text{eclose}(s) = \text{set of all states reachable from } s \text{ in zero or more epsilon transitions}$

**$M'$ components**
$s$ is an accepting state of $M'$ iff $\text{eclose}(s)$ contains an accepting state

$s \xrightarrow{c} t$ is a transition in $M'$ iff $q \xrightarrow{c} t$ for some $q$ in $\text{eclose}(s)$
**Def:** $\text{eclose}(s) = \text{set of all states reachable from } s \text{ in zero or more epsilon transitions}$

$s$ is an accepting state of $M'$ iff $\text{eclose}(s)$ contains an accepting state.

$s \xrightarrow{c} t$ is a transition in $M'$ iff $q \xrightarrow{c} t$ for some $q$ in $\text{eclose}(s)$.
Recap

NFAs and DFAs are equally powerful

- any language definable as an NFA is definable as a DFA
- $\varepsilon$-transitions do not add expressiveness to NFAs
- we showed a simple algorithm to remove epsilons
Regular expressions

Pattern describing a language

**operands:** single characters, epsilon

**operators:** from low to high precedence

alternation “or”:  a | b

catenation:  a.b,  ab,  a^3 (which is aaa)

iteration:  a* (0 or more a’s) aka Kleene star
Regexp, cont’d

Conventions:

a+ is a.a*
ladder is a|b|c|d|…|y|z|A|B|…|Z
digit is 0|1|2|…|9
not(x) all characters except x
. is any character
parentheses for grouping, e.g., (ab)*
ε, ab, abab, ababab
Regexp, example

Hex strings

- start with 0x or 0X
- followed by one or more hexadecimal digits
- optionally end with l or L

$$0(x|X)\text{hexdigit}+(L|l|\varepsilon)$$

where hexdigit = digit|a|b|c|d|e|f|A|…|F

OR:

$$(0(x|X)\text{hexdigit}_\text{lowercase}+(L|l|\varepsilon)) \mid (0(x|X)\text{hexdigit}_\text{uppercase}+(L|l|\varepsilon))$$
Regexp, example

Single-line comments in Java/C/C++

    // this is a comment
    //((not(\n))*\n