Regular Expressions

CS 2800: Discrete Structures, Fall 2014

Sid Chaudhuri

Regular expressions ("regex"-es) are defined *inductively*

(start with simple base expressions, construct more complicated ones recursively)

Empty set

$$L(\emptyset) = \emptyset$$

Empty string

$$L(\boldsymbol{\varepsilon}) = \{\varepsilon\}$$

Literal character

$$L(\mathbf{x}) = \{x\}$$

e.g. $L(\mathbf{1}) = \{1\}, L(\mathbf{2}) = \{2\}, L(\mathbf{a}) = \{a\}$

Concatenation

$$L(AB) = \{ab \mid a \in L(A), \mathrm{b} \in L(B)\}$$

e.g. $L(\mathbf{12}) = \{12\}, L(\mathbf{aabb}) = \{aabb\}$

Concatenation

$$L(AB) = \{ab \mid a \in L(A), \mathrm{b} \in L(B)\}$$

e.g. $L(\mathbf{12}) = \{12\}$, $L(\mathbf{aabb}) = \{aabb\}$,
$L(\mathbf{a\varepsilon}) = ?$

Concatenation

$$L(AB) = \{ab \mid a \in L(A), \mathrm{b} \in L(B)\}$$

e.g. $L(\mathbf{12}) = \{12\}$, $L(\mathbf{aabb}) = \{aabb\}$,
$L(\mathbf{a\varepsilon}) = \{a\}$

Concatenation

$$L(AB) = \{ab \mid a \in L(A), \mathrm{b} \in L(B)\}$$

e.g. $L(\mathbf{12}) = \{12\}$, $L(\mathbf{aabb}) = \{aabb\}$,
$L(\mathbf{a\varepsilon}) = \{a\}$, $L(\mathbf{a}\varnothing) = ?$

Concatenation

$$L(AB) = \{ab \mid a \in L(A), \mathrm{b} \in L(B)\}$$

e.g. $L(\mathbf{12}) = \{12\}, L(\mathbf{aabb}) = \{aabb\},$
$L(\mathbf{a\varepsilon}) = \{a\}, L(\mathbf{a}\varnothing) = \varnothing$

Alternation

$$L(A|B) = L(A) \cup L(B)$$

e.g. $L(\mathbf{1}|\mathbf{2}) = \{1, 2\}$, $L(\mathbf{aa}|\mathbf{bb}) = \{aa, bb\}$

Alternation

$$L(A|B) = L(A) \cup L(B)$$

e.g. $L(\mathbf{1}|\mathbf{2}) = \{1, 2\}$, $L(\mathbf{aa}|\mathbf{bb}) = \{aa, bb\}$,
$L(\mathbf{a}|\boldsymbol{\varepsilon}) = ?$

Alternation

$$L(A|B) = L(A) \cup L(B)$$

e.g. $L(\mathbf{1}|\mathbf{2}) = \{1, 2\}$, $L(\mathbf{aa}|\mathbf{bb}) = \{aa, bb\}$,
$L(\mathbf{a}|\boldsymbol{\varepsilon}) = \{a, \varepsilon\}$

Alternation

$$L(A|B) = L(A) \cup L(B)$$

e.g. $L(\mathbf{1}|\mathbf{2}) = \{1, 2\}$, $L(\mathbf{aa}|\mathbf{bb}) = \{aa, bb\}$,
$L(\mathbf{a}|\boldsymbol{\varepsilon}) = \{a, \varepsilon\}$, $L(\mathbf{a}|\emptyset) = ?$

Alternation

$$L(A|B) = L(A) \cup L(B)$$

e.g. $L(\mathbf{1}|\mathbf{2}) = \{1, 2\}$, $L(\mathbf{aa}|\mathbf{bb}) = \{aa, bb\}$,
$L(\mathbf{a}|\boldsymbol{\varepsilon}) = \{a, \varepsilon\}$, $L(\mathbf{a}|\emptyset) = \{a\}$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a^*}) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = ?$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$,
$L((\mathbf{a}|\mathbf{b})^*) = ?$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\ x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$,
$L((\mathbf{a}|\mathbf{b})^*) = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, \ldots\}$

Kleene star

$$L(A^*) = \{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\ x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$,
$L((\mathbf{a}|\mathbf{b})^*) = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, \ldots\}$,
$L(\boldsymbol{\varepsilon}^*) = ?$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N}, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$,
$L((\mathbf{a}|\mathbf{b})^*) = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, \ldots\}$,
$L(\boldsymbol{\varepsilon}^*) = \{\varepsilon\}$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N}, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$,
$L((\mathbf{a}|\mathbf{b})^*) = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, \ldots\}$,
$L(\boldsymbol{\varepsilon}^*) = \{\varepsilon\}$, $L(\varnothing^*) = ?$

Kleene star

$$L(A^*) =$$
$$\{\varepsilon\} \cup \{\, x_1 x_2 \ldots x_n \mid n \in \mathbf{N},\, x_i \in L(A) \,\}$$

e.g. $L(\mathbf{a}^*) = \{\varepsilon, a, aa, aaa, aaaa, \ldots\}$,
$L((\mathbf{ab})^*) = \{\varepsilon, ab, abab, ababab, \ldots\}$,
$L((\mathbf{a}|\mathbf{b})^*) = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, \ldots\}$,
$L(\boldsymbol{\varepsilon}^*) = \{\varepsilon\}$, $L(\varnothing^*) = \{\varepsilon\}$

Playing with regexes

- http://regex101.com/
- http://rubular.com/
- http://www.google.com/search?q=online+regex+tester

Kleene's Theorem

- A language is recognized by a regular expression (that is, it is a “regular language”) if and only if it is recognized by a finite automaton

Kleene's Theorem

- A language is recognized by a regular expression (that is, it is a “regular language”) if and only if it is recognized by a finite automaton
  - Regex has FA
    - Relatively simple construction

Kleene's Theorem

- A language is recognized by a regular expression (that is, it is a "regular language") if and only if it is recognized by a finite automaton
  - Regex has FA
    - Relatively simple construction
  - FA has regex
    - Tricky to prove

Regex → FA

- For every regular expression, there is a finite automaton that recognizes the same language

Regex → FA

- For every regular expression, there is a finite automaton that recognizes the same language
- We will construct an ε-NFA

Regex → FA

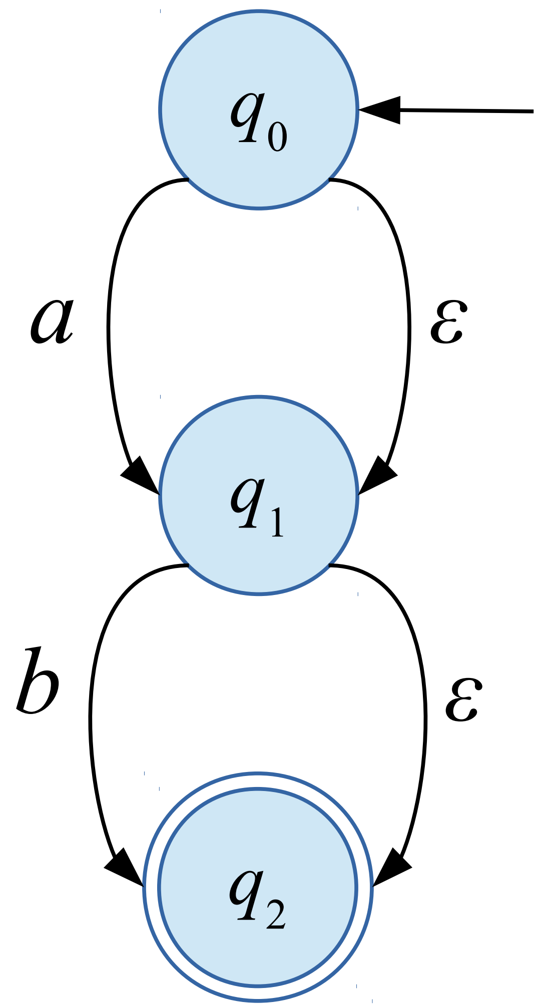
- For every regular expression, there is a finite automaton that recognizes the same language
- We will construct an ε-NFA
  - ... which can be converted to an NFA

Regex → FA

- For every regular expression, there is a finite automaton that recognizes the same language
- We will construct an ε-NFA
  - … which can be converted to an NFA
  - … which can be converted to a DFA

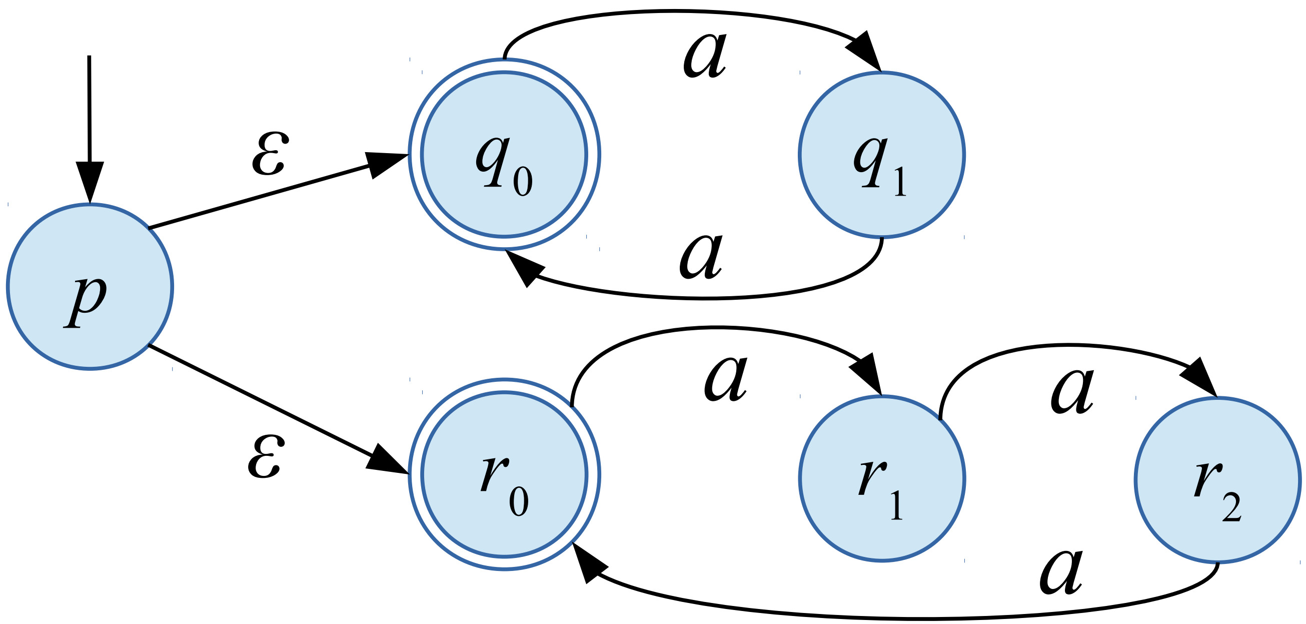
Recap: NFAs with epsilon transitions

- Just like ordinary NFAs, but…
  - Can "instantaneously" change state *without* reading an input symbol
  - Valid transitions of this type are shown by arcs labeled 'ε'
  - Note that ε does not suddenly become a member of the alphabet. Instead, we assume ε does not belong to *any* alphabet – it's a special symbol.



Why ε-NFAs?

- Suitable for representing "or" relations
- E.g. $L = \{ \, a^n \mid n \in \mathbf{N} \text{ is divisible by 2 or 3} \, \}$



- ... but they're equivalent to NFAs and DFAs