Theory of Computation

Tutorial - Regular Expressions

Cesare Spinoso-Di Piano

Plan for today

1. Regular expressions
2. Regular Expressions to FAs
3. FAs to Regular Expressions

Regular expressions

Regular expressions

Definition. Given an alphabet Σ, define a regular expression as follows:

1. $\emptyset, \lambda, a \in \Sigma$ are the **primitive** regular expressions.
2. If r_1 and r_2 are regular expressions, then so are $r_1 + r_2$, $r_1 \cdot r_2$, r_1^* and (r_1).
3. A string s is a regular expression **if and only if** it can be obtained by applying operations in 2. to the primitive regular expressions in 1.

Examples.

- **Example.** $\Sigma = \{a, b\}$, $(a + \lambda) \cdot (b \cdot a + \emptyset^*)$ is a valid regular expression.
- **Example.** $\Sigma = \{a, b\}$, $(a \cap \lambda) - (b^R \cdot \emptyset)$ is NOT a valid regular expression.

Regular expressions

Definition. Given the regular expressions r, $L(r)$ is the language **denoted by/represented by** the regular expression r where

$L(\emptyset) = \emptyset$, $L(\lambda) = \{\lambda\}$, $L(a) = \{a\}, a \in \Sigma$

If r_1, r_2 are regular expressions

$$L(r_1 + r_2) = L(r_1) \cup L(r_2)$$
$$L(r_1 r_2) = L(r_1)L(r_2)$$
$$L(r_1^*) = (L(r_1))^*$$

Example

Example. Given the regular expressions $r = (a + bb)(ab)^*$, what is $L(r)$?

$$\begin{aligned}
L(r) &= L((a + bb)(ab)^*) \\
&= L(a + bb)L((ab)^*) && \text{Using 3.} \\
&= \Big(L(a) \cup L(bb)\Big)\Big(L(ab)\Big)^* && \text{Using 2. and 4.} \\
&= \Big(\{a\} \cup \{bb\}\Big)\Big(\{ab\}\Big)^* && \text{Using 1.} \\
&= \{a, bb\}\{ab\}^* \\
&= \{a(ab)^n : n \geq 0\} \cup \{bb(ab)^n : n \geq 0\}
\end{aligned}$$

Exercise

Exercise. Given the regular expressions $r = (0 + 01)(1 + 11)^*$, how many strings of length **3 or less** are there in $L(r)$?

Exercise

Exercise. What is the the shortest string in $L(((a+\lambda)\emptyset^* + b + \emptyset)^*) \cap L(a(ba)^*)$?

Exercise

Exercise. Write a regular expression r such that $L(r) = \{w \in \{0,1\}^* : \text{w contains } 101\}$.

Exercise

Exercise. Write a regular expression r such that
$L(r) = \{w \in \{0,1\}^* : \text{w has even length}\}$.

Exercise

Exercise. Write a regular expression r such that $L(r) = \{vwv : |v| = 2, v, w \in \{a, b\}^*\}$.

Equivalent regular expressions

Definition. Two regular expressions r_1, r_2 are equivalent $(r_1 \equiv r_2)$ if and only if $L(r_1) = L(r_2)$.

Example. True or False. $(r + \emptyset)^* \lambda = r^*$.
This is True since

$$\begin{aligned} L((r + \emptyset)^* \lambda) &= L((r + \emptyset)^*) L(\lambda) \\ &= (L(r + \emptyset))^* L(\lambda) \\ &= (L(r) \cup \emptyset)^* L(\lambda) \\ &= (L(r))^* \\ &= L(r^*) \end{aligned}$$

In general, regular expression "arithmetic" is avoided in favor of FA manipulations.

Regular Expressions to FAs

Regular expressions to FAs

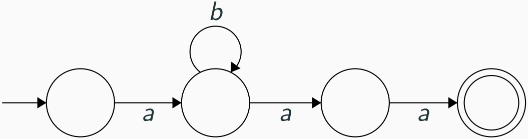
Theorem. Given a regular expression r, there exists an NFA N such that $L(M) = L(r)$.

Corollary. Given a regular expression r, there exists an FA M such that $L(M) = L(r)$.

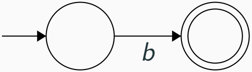
Example

Example. Given $r = ab^*aa + b(ba)^*$, find an FA M such that $L(M) = L(r)$.

1. Split the language into components. Here we have $L(ab^*aa)$, $L(b)$, $L((ba)^*)$.
2. Draw an FA for each regular language separately.
   An FA for $L(ab^*aa)$.

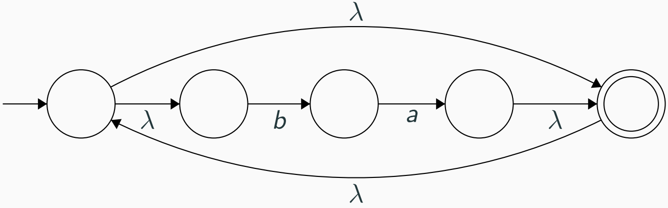


   An FA for $L(b)$.

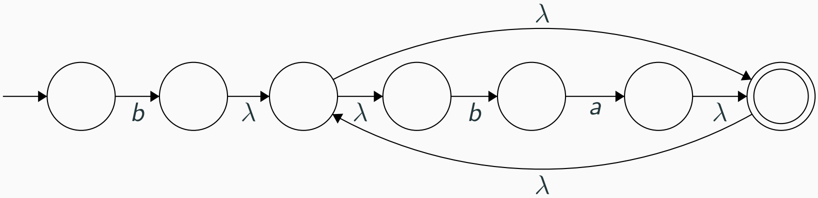


Example

An FA for $L((ba)^*)$.

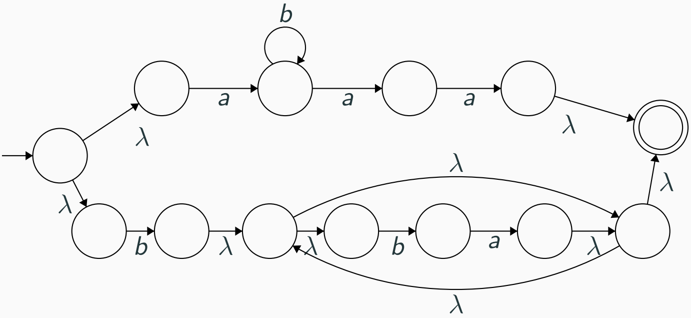


An FA for $L(b(ba)^*)$.



Example

Combining all these machines together.



Exercise

Exercise. Create an NFA and a DFA that accept $L((abb)^* + (a^* bb^*))$.

FAs to Regular Expressions

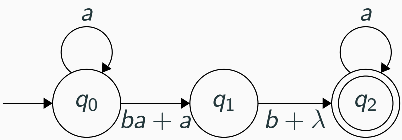
Generalized Transition Graph

Theorem. Given a finite automaton M, there exists a regular expressions r such that $L(r) = L(M)$.

How can we convert a finite automata to a regular expression?
Generalized Transition Graphs.

Definition. A generalized transition graph is a transition graph whose edges are labeled with regular expressions. Therefore, the transition $\delta'(q, r)$ executes if the GTG reads any string that belongs to $L(r)$.

Example. The following GTG is of an FA that accepts $L(a^*(ba + a)(b + \lambda)a^*)$.

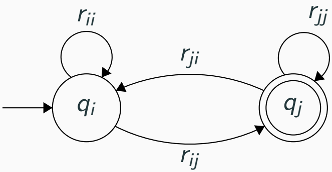


FA to RegExp

How can we convert a finite automata to a regular expression using GTGs?

1. Convert the FA to an NFA with a single final state (which should be distinct from q_0).
2. Convert the FA to a GTG.
3. Remove each intermediate state while preserving its role in the GTG.
4. Repeat until only the initial and final states are left.

The resulting GTG should have the following form.

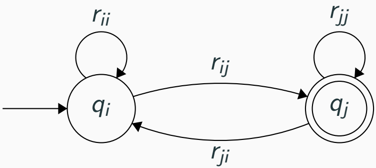


Then we see that this GTG reduces to the regular expression:
$r = r_{ii}{}^* r_{ij}(r_{jj} + r_{ji} r_{ii}{}^* r_{ij})^*$

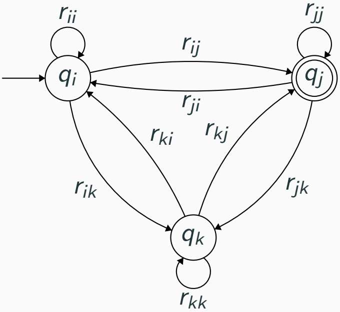
FA to RegExp

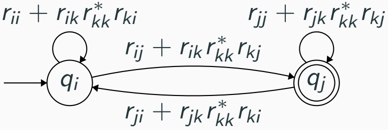
A more detailed explanation of the algorithm.

1. Convert the FA to an NFA with a single final state (which should be distinct from q_0).
2. Convert the NFA into a generalized transition graph. Let r_{ij} stand for the label (a regular expression) of the edge from q_i to q_j.
3. If the GTG only has the initial state q_i and final state q_j, compute the final regular expression $r = {r_{ii}}^* r_{ij}(r_{jj} + r_{ji}{r_{ii}}^* r_{ij})^*$.



4. If the GTG has three states, the initial q_i state, the final state q_j and third state q_k. Introduce new edges, labeled $r_{pq} + r_{pk} r_{kk}{}^* r_{kq}$, for $p = i, j, q = i, j$. When this is done, remove q_k and its associated edges.





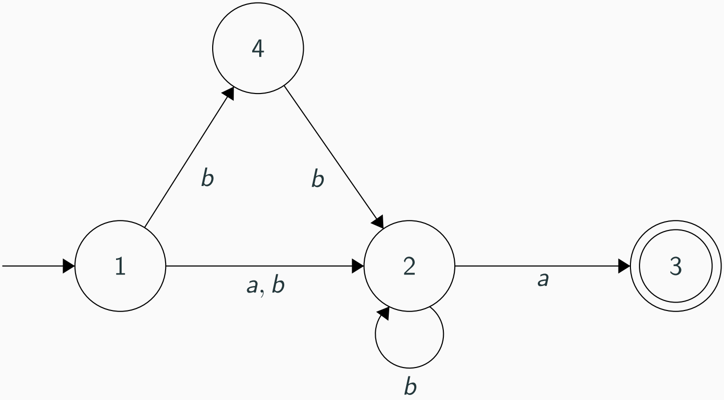
FA to RegExp

5. If the GTG has more than three states, pick a state q_k, apply the equation in step 4 for all pairs of states (q_i, q_j), $i \neq k, j \neq k$.

6. Repeat step 2 to 4 until the GTG has only an initial and final state and then apply $r = {r_{ii}}^* r_{ij}(r_{jj} + r_{ji}{r_{ii}}^* r_{ij})^*$.

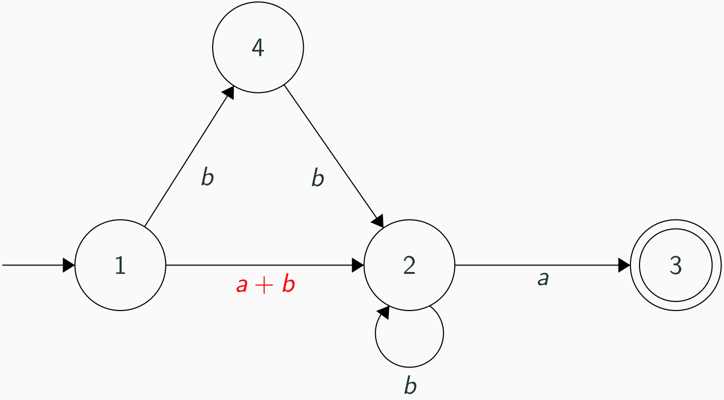
Example

Example. Find a regular expression that **denotes** the language **accepted** by the following NFA.



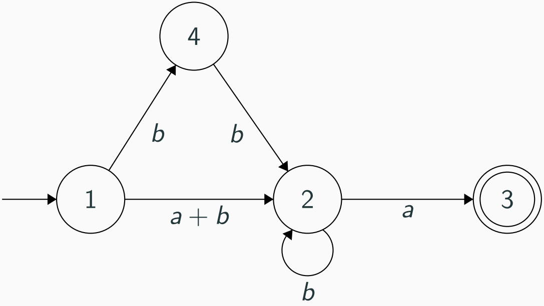
Example

The equivalent GTG.



Example

Pick a state to remove. Here we start with state 2.

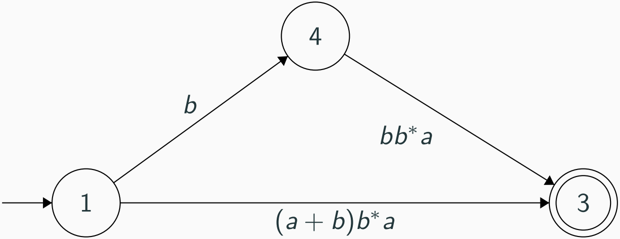


Observe in what way 2 acts as an intermediate state.

- From 1 to 3: $(a + b)b^*a$
- From 4 to 3: bb^*a

Example

Remove state 2 and its associated edges. Add to edges 1 to 3 and 4 to 3 the intermediate regular expressions.



Example

Pick state 4 to remove and find how it acts as an intermediate state.

- From 1 to 3: $(a+b)b^*a + bbb^*a$

Remove state 4 and its associated edges. Add to edges 1 to 3 the intermediate regular expression.

$$
\rightarrow (1) \xrightarrow{(a+b)b^*a + bbb^*a} ((3))
$$

Obtain the final regular expression: $(a+b)b^*a + bbb^*a$

Exercise

Exercise. Find a regular expression that denotes the language
$L = \{w \in \{0,1\}^* : n_0(w) \mod 2 = 0 \;\&\; n_1(w) \mod 2 = 0\}$

Regular expressions and regular languages

Theorem. The family of languages accepted by regular expressions is exactly the same as the family of languages accepted by FAs. Or, in other words, a language L is regular if and only if there exists a regular expression r such that $L = L(r)$.