Finite Automata and Formal Languages

TMV027/DIT321– LP4 2013

Lecture 8
Ana Bove

April 23rd 2013

Overview of today's lecture:

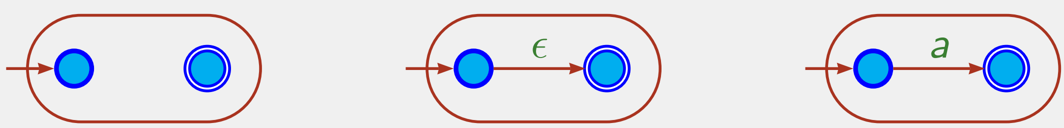
- Equivalence between FA and RE: from RE to FA;
- Pumping Lemma for RL;
- Closure properties of RL.

From Regular Expressions to Finite Automata

Proposition: *Every language defined by a RE is accepted by a FA.*

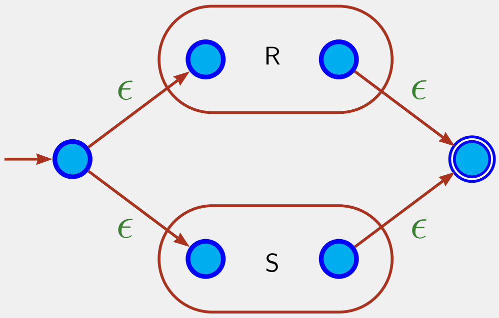
Proof: Let $\mathcal{L} = \mathcal{L}(R)$ for some RE R. By induction on R we construct a ϵ-NFA E with only one final state and no arcs into the initial state or out of the final state, and such that $\mathcal{L} = \mathcal{L}(E)$.

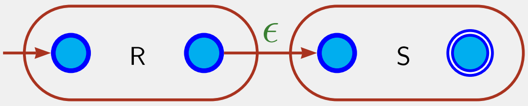
Base cases are $\emptyset$, ϵ and $a \in \Sigma$. The corresponding ϵ-NFA recognising the languages $\emptyset$, $\{\epsilon\}$ and $\{a\}$ respectively, are:

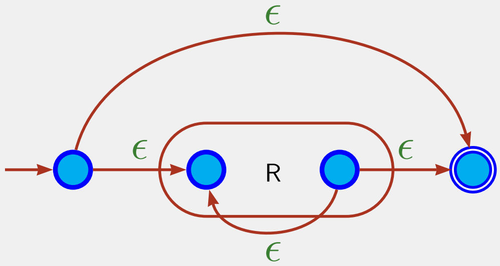


From RE to FA: Inductive Step

Given the RE R and S and FA for them, we construct the FA for $R + S$, RS and R^* recognising the languages $\mathcal{L}(R) \cup \mathcal{L}(S)$, $\mathcal{L}(R)\mathcal{L}(S)$ and $\mathcal{L}(R)^*$ respectively:

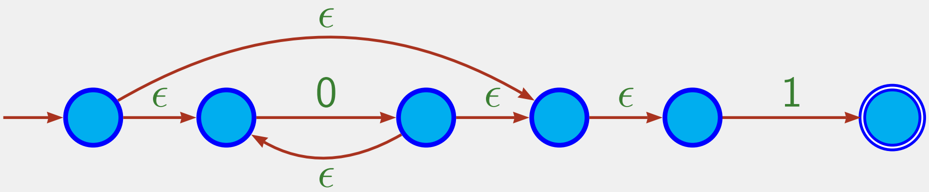




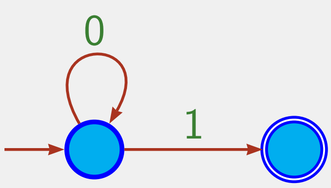


Example: From RE to FA

Let us follow this method to construct a FA for the RE 0^*1.



Compare it with the following FA:



How to Identify Regular Languages?

We have seen that a language is regular iff there is a DFA that accepts the language.

Then we saw that DFA, NFA and ϵ-NFA are equivalent in the sense that we can convert between them.

Hence FA accept all and only the regular languages (RL).

Now we have seen how to convert between FA and RE.

Thus RE also define all and only the RL.

How to Prove that a Language is NOT Regular?

In a FA with n states, any path

$$q_1 \xrightarrow{a_1} q_2 \xrightarrow{a_2} q_3 \xrightarrow{a_3} \ldots \xrightarrow{a_{m-1}} q_m \xrightarrow{a_m} q_{m+1}$$

has a loop if $m \geqslant n$.

That is, we have $i < j$ such that $q_i = q_j$ in the path above.

This can be seen as an application of the *Pigeonhole Principle*, which is an important reasoning technique in mathematics and computer science.

(See Wikipedia.)

How to Prove that a Language is NOT Regular?

Example: Let us prove that $\mathcal{L} = \{0^m 1^m | m \geqslant 0\}$ is not a RL.

Let us assume it is: then $\mathcal{L} = \mathcal{L}(A)$ for some FA A with n states, $n > 0$.

Let $k \geqslant n > 0$ and let $w = 0^k 1^k \in \mathcal{L}$.
Then there must be an accepting path $q_0 \xrightarrow{w} q \in F$.

Since $k \geqslant n$, there is a loop (pigeonhole principle) when reading the 0's.

Then $w = xyz$ with $|xy| = j \leqslant n$, $y \neq \epsilon$ and $z = 0^{k-j} 1^k$ such that

$$q_0 \xrightarrow{x} q_l \xrightarrow{y} q_l \xrightarrow{z} q \in F$$

Observe that the following path is also an accepting path

$$q_0 \xrightarrow{x} q_l \xrightarrow{z} q \in F$$

However y must be of the form 0^i with $i > 0$ hence $xz = 0^{k-i} 1^k \notin \mathcal{L}$.

This contradicts the fact that A accepts $\mathcal{L}$.

The Pumping Lemma for Regular Languages

Theorem: *Let $\mathcal{L}$ be a RL. Then, there exists a constant n—which depends on $\mathcal{L}$—such that for every string $w \in \mathcal{L}$ and $|w| \geqslant n$, we can break w into 3 strings x, y and z such that $w = xyz$ and*

1. $y \neq \epsilon$
2. $|xy| \leqslant n$
3. $\forall k \geqslant 0, xy^k z \in \mathcal{L}$

Proof of the Pumping Lemma

Assume we have a FA A that accepts the language, then $\mathcal{L} = \mathcal{L}(A)$.

Let n be the number of states in A.

Then any path of length $m \geqslant n$ has a loop.

Let us consider $w = a_1 a_2 \ldots a_m \in \mathcal{L}$.

We have an accepting path and a loop such that

$$q_0 \xrightarrow{x} q_l \xrightarrow{y} q_l \xrightarrow{z} q \in F$$

with $w = xyz \in \mathcal{L}$, $y \neq \epsilon$, $|xy| \leqslant n$.

Then we also have

$$q_0 \xrightarrow{x} q_l \xrightarrow{y^k} q_l \xrightarrow{z} q \in F$$

for any k, that is, $\forall k \geqslant 0, xy^k z \in \mathcal{L}$.

Example: Application of the Pumping Lemma

Let us use the Pumping lemma to prove that $\{0^m 1^m | m \geqslant 0\}$ is not a RL.

We assume it is.

Let n be the constant given by the lemma and let $w = 0^n 1^n$, then $|w| \geqslant n$.

By the lemma we know that $w = xyz$ with $y \neq \epsilon$, $|xy| \leqslant n$ and $\forall k \geqslant 0, xy^k z \in \mathcal{L}$.

Since $y \neq \epsilon$ and $|xy| \leqslant n$, we know that $y = 0^i$ with $i \geqslant 1$.

However, we have a contradiction since $xy^k z \notin \mathcal{L}$ for $k \neq 1$.

Note: The Pumping lemma is connected to the fact that a FA has *finite memory*! If we could build a machine with infinitely many states it would be able to recognise the language.

Example: Application of the Pumping Lemma

Example: Let us prove that $\mathcal{L} = \{0^i 1^j | i \leqslant j\}$ is not a RL.

Let n be given by the Pumping lemma and let $w = 0^n 1^{n+1} \in \mathcal{L}$, hence $|w| \geqslant n$.

Then we know that $w = xyz$ with $y \neq \epsilon$, $|xy| \leqslant n$ and $\forall k \geqslant 0, xy^k z \in \mathcal{L}$.

Since $y \neq \epsilon$ and $|xy| \leqslant n$, we know that $y = 0^r$ with $r \geqslant 1$.

However, we have a contradiction since $xy^k z \notin \mathcal{L}$ for $k > 2$.

(Even for $k = 2$ if $r > 1$.)

Example: What about the languages $\{0^i 1^j \mid i \geqslant j\}$, $\{0^i 1^j \mid i > j\}$ and $\{0^i 1^j \mid i \neq j\}$?
Does the Pumping lemma help?

Pumping Lemma is not a Necessary Condition

By showing that the Pumping lemma does not apply to a certain language $\mathcal{L}$ we prove that $\mathcal{L}$ is not regular.

However, if the Pumping lemma *does* apply to $\mathcal{L}$, we *cannot* conclude whether $\mathcal{L}$ is regular or not!

Example: We know $\mathcal{L} = \{b^m c^m \mid m \geqslant 0\}$ is not regular.

Let us consider $\mathcal{L}' = a^+ \mathcal{L} \cup (b + c)^*$.

$\mathcal{L}'$ is not regular. If $\mathcal{L}'$ would be regular, then we can prove that $\mathcal{L}$ is regular (using the closure properties we will see next).

However, the Pumping lemma does apply for $\mathcal{L}'$ with $n = 1$.

This shows the Pumping lemma is not a necessary condition for a language to be regular.

Closure Properties for Regular Languages

Let $\mathcal{M}$ and $\mathcal{N}$ be RL. Then $\mathcal{M} = \mathcal{L}(R) = \mathcal{L}(D)$ and $\mathcal{N} = \mathcal{L}(S) = \mathcal{L}(F)$ for RE R and S, and DFA D and F.

We have seen that RL are closed under the following operations:

Union: $\mathcal{M} \cup \mathcal{N} = \mathcal{L}(R + S)$ or $\mathcal{M} \cup \mathcal{N} = \mathcal{L}(D \oplus F)$ (s.20, l.4);

Complement: $\overline{\mathcal{M}} = \mathcal{L}(\overline{D})$ (slide 21, lec. 4)

Intersection: $\mathcal{M} \cap \mathcal{N} = \overline{\overline{\mathcal{M}} \cup \overline{\mathcal{N}}}$ or $\mathcal{M} \cap \mathcal{N} = \mathcal{L}(D \times F)$ (s.19, l.4);

Difference: $\mathcal{M} - \mathcal{N} = \mathcal{M} \cap \overline{\mathcal{N}}$;

Concatenation: $\mathcal{M}\mathcal{N} = \mathcal{L}(RS)$;

Closure: $\mathcal{M}^* = \mathcal{L}(R^*)$.

More Closure Properties for Regular Languages

We shall now see that RL are also closed under the following operations:

Prefix: See exercise 2 on DFA. Hint: in D, make final all states in a path from the start state to final state.

Reversal: Recall that $\text{rev}(a_1 \ldots a_n) = a_n \ldots a_1$ and $\forall x, \text{rev}(\text{rev}(x)) = x$ (slides 34 & 36, lec. 2);

Homomorphism: Structure-preserving functions between 2 algebraic structures (slide 29, lec. 2). Recall languages are monoids. We will map symbols to strings. Intuitively, $h(a_1 \ldots a_n) = h(a_1) \ldots h(a_n)$.

Inverse homomorphism:

Closure under Prefix

Another way to prove that the language of prefixes of a RL is regular is as follows.

Define the following function over RE:

$$\begin{aligned} &pre(\emptyset) = \emptyset \\ &pre(\epsilon) = \epsilon \\ &pre(a) = \epsilon + a \\ &pre(R_1 + R_2) = pre(R_1) + pre(R_2) \\ &pre(R_1 R_2) = pre(R_1) + R_1 pre(R_2) \\ &pre(R^*) = R^* pre(R) \end{aligned}$$

and prove that $\mathcal{L}(pre(R)) = \text{Prefix}(\mathcal{L}(R))$.

Then, if $\mathcal{M} = \mathcal{L}(R)$ for some RE R then
$\text{Prefix}(\mathcal{M}) = \text{Prefix}(\mathcal{L}(R)) = \mathcal{L}(pre(R))$.

Closure under Reversal

We define the following function over RE:

$$\begin{aligned} &\emptyset^{\mathsf{r}} = \emptyset \qquad \epsilon^{\mathsf{r}} = \epsilon \qquad a^{\mathsf{r}} = a \\ &(R_1 + R_2)^{\mathsf{r}} = R_1^{\mathsf{r}} + R_2^{\mathsf{r}} \\ &(R_1 R_2)^{\mathsf{r}} = R_2^{\mathsf{r}} R_1^{\mathsf{r}} \\ &(R^*)^{\mathsf{r}} = (R^{\mathsf{r}})^* \end{aligned}$$

Theorem: *If $\mathcal{M}$ is regular so is $\mathcal{M}^{\mathsf{r}}$.*

Proof: (See theo. 4.11, pages 139–140).

Let R be a RE such that $\mathcal{M} = \mathcal{L}(R)$.
We need to prove by structural induction on R that $\mathcal{L}(R^{\mathsf{r}}) = (\mathcal{L}(R))^{\mathsf{r}}$.
Hence $\mathcal{M}^{\mathsf{r}} = (\mathcal{L}(R))^{\mathsf{r}} = \mathcal{L}(R^{\mathsf{r}})$ and $\mathcal{M}^{\mathsf{r}}$ is regular.

Example: The reverse of the language defined by $(0+1)^*0$ can be defined by $0(0+1)^*$.

Closure under Reversal

Another way to prove this result is by constructing a ϵ-NFA for $\mathcal{M}^{\mathsf{r}}$.

Proof: Let $N = (Q, \Sigma, \delta_N, q_0, F)$ be a NFA such that $\mathcal{M} = \mathcal{L}(N)$.

Define $E = (Q \cup \{q\}, \Sigma, \delta_E, q, \{q_0\})$ with $q \notin Q$ and δ_E such that

$$r \in \delta_E(s, a) \text{ iff } s \in \delta_N(r, a) \text{ for } r, s \in Q$$
$$r \in \delta_E(q, \epsilon) \text{ iff } r \in F$$

Closure under Homomorphisms

Theorem: *If $\mathcal{M}$ is a RL over Σ and $h : \Sigma^* \to \Delta^*$ is an homomorphism on Σ then $h(\mathcal{M})$ is also regular.*

Proof: We define the following function over RE:

$$f_h(\emptyset) = \emptyset \qquad f_h(\epsilon) = \epsilon \qquad f_h(a) = h(a)$$
$$f_h(R_1 + R_2) = f_h(R_1) + f_h(R_2)$$
$$f_h(R_1 R_2) = f_h(R_1) f_h(R_2)$$
$$f_h(R^*) = (f_h(R))^*$$

We need to prove by structural induction on R that $\mathcal{L}(f_h(R)) = h(\mathcal{L}(R))$.

Now, if $\mathcal{M} = \mathcal{L}(R)$ then we have that $h(\mathcal{M})$ is regular since $h(\mathcal{M}) = h(\mathcal{L}(R)) = \mathcal{L}(f_h(R))$.

(See Theorem 4.14, pages 141–142.)

Closure under Homomorphisms

Let $h : \Sigma^* \to \Delta^*$ be a homomorphism and $\mathcal{M}$ a RL over Σ.

By the previous theorem and the definition of RL, we know that there exists a DFA D over Σ and a DFA F over Δ such that

$$\mathcal{M} = \mathcal{L}(D) \quad \text{and} \quad h(\mathcal{M}) = \mathcal{L}(F)$$

F can be constructed from the RE for $\mathcal{M}$ (via an ϵ-NFA).

Often not obvious how to construct the DFA directly.

Inverse Homomorphisms

Definition: If $h : \Sigma^* \to \Delta^*$ is a homomorphism and $\mathcal{L}$ is a language over Δ, $h^{-1}(\mathcal{L})$ (read *h inverse of* $\mathcal{L}$) is the set of strings w such that $h(w) \in \mathcal{L}$.

In other words, $h^{-1}(\mathcal{L}) = \{w \in \Sigma^* \mid h(w) \in \mathcal{L}\}$.

Note: h^{-1} does not necessarily correspond to a function!

Example: Imagine we have that $h(a) = c$, $h(b) = c$ and $\mathcal{L} = \{c\}$.

Then $h^{-1}(\mathcal{L}) = \{a, b\}$ but h^{-1} itself is not a function.

Closure under Inverse Homomorphisms

Theorem: *Let $h : \Sigma^* \to \Delta^*$ be a homomorphism. If $\mathcal{M}$ is a RL over Δ then $h^{-1}(\mathcal{M})$ is a RL over Σ.*

Proof: Let $D = (Q, \Delta, \delta, q_0, F)$ be a DFA such that $\mathcal{M} = \mathcal{L}(D)$.

We define the DFA $D' = (Q, \Sigma, \delta', q_0, F)$ over Σ such that

$$\delta'(q, a) = \hat{\delta}(q, h(a))$$

By induction on $|w|$ we prove that $\hat{\delta}'(q, w) = \hat{\delta}(q, h(w))$

(Recall that $\hat{\delta}(q, xy) = \hat{\delta}(\hat{\delta}(q, x), y)$.)

Then D' accepts w iff D accepts $h(w)$ (since the set of accepting states is the same in both DFA).

Note: Since h^{-1} might not be a function it seems difficult to directly define the RE that corresponds to the h inverse of $\mathcal{M}$.

Example: $\mathcal{L}'$ from Slide 11

Example: We know $\mathcal{L} = \{b^m c^m \mid m \geqslant 0\}$ is not regular.

Let us consider $\mathcal{L}' = a^+\mathcal{L} \cup (b + c)^*$.

We will prove that $\mathcal{L}'$ is not regular. Let us assume it is.

Then $a^+\mathcal{L} = \mathcal{L}' \cap \overline{(b + c)^*}$ must be regular.

Then, $\mathcal{L} = h(a^+\mathcal{L})$ must also be regular, where h is the following homomorphism: $h(a) = \epsilon, h(b) = b, h(c) = c$.

We arrive at a contradiction, hence $\mathcal{L}'$ cannot be regular.

Overview of Next Lecture

Sections 4.3–4.4:

- Decision Properties for RL;
- Equivalence of RL;
- Minimisation of automata.