Finite Automata and Regular Languages (part VI)

Prof. Dan A. Simovici

UMB

For any regular language L there are several automata that are capable of recognizing it. Naturally, we are interested in finding among these automata the ones that have the smallest number of states.

Definition

The *Nerode equivalence* of a language $L \subseteq A^*$ is the relation:

$$\nu_L = \{(x, y) \in A^* \times A^* \mid xw \in L \text{ if and only if } yw \in L \text{ for every } w \in A^*\}.$$

The relation ν_L is a right-invariant equivalence relation.
In other words: if $(x, y) \in \nu_L$, then $(xu, yu) \in \nu_L$ for every $u \in A^*$. In terms of equivalence classes, $[x]_{\nu_L} = [y]_{\nu_L}$ implies $[xu]_{\nu_L} = [yu]_{\nu_L}$ for every $u \in A^*$.
Recall that $[x]_{\nu_L}$ denotes the ν_L-equivalence class of the word x. The set of all equivalence classes of ν_L will be denoted by A^*/ν_L.

Lemma

Let $L \subseteq A^*$ *be a language over an alphabet* A*. We have* $(x, y) \in \nu_L$ *if and only if* $x^{-1}L = y^{-1}L$*.*

Proof

Let $x, y \in A^*$ such that $(x, y) \in \nu_L$, and let $t \in x^{-1}L$. This means that $xt \in L$, which implies $yt \in L$ because of the definition of ν_L. Therefore, $t \in y^{-1}L$, so $x^{-1}L \subseteq y^{-1}L$. The reverse inclusion can be obtained in the same manner, so $x^{-1}L = y^{-1}L$.
Conversely, if $x^{-1}L = y^{-1}L$, then $xt \in L$ if and only if $yt \in L$ for every $t \in A^*$, which means that $(x, y) \in \nu_L$.

Definition

Let $L \subseteq A^*$ be a language over the alphabet A. The *set of left derivatives of L* is the set $\mathcal{Q}_L = \{t^{-1}L \mid t \in A^*\}$.

Lemma

Let $L \subseteq A^*$ be a language over an alphabet A. The set of left derivatives of L is finite if and only if A^*/ν_L is finite.

Proof.

The function $h_L : A^*/\nu_L \longrightarrow \mathcal{Q}_L$ defined by $h_L([x]_{\nu_L}) = x^{-1}L$ is a bijection. The desired conclusion follows immediately. □

Lemma

Any language $L \subseteq A^*$ is a ν_L-saturated set.

Proof.

In order to prove that L is ν_L-saturated, it suffices to show that the ν_L-equivalence class of every $x \in L$ is included in L. Let $x \in L$. If $(x, y) \in \nu_L$, then $yz \in L$ whenever $xz \in L$ for any $z \in A^*$. Selecting $z = \lambda$ gives the required result. □

- Note that the previous lemma is equivalent to saying that for all words $x, y \in A^*$, if $x \in L$ and $x^{-1}L = y^{-1}L$, then $y \in L$.
- No assumption is made about the language L; in particular, L need not be regular.

Definition

Let $L \subseteq A^*$ be a language over the alphabet A.
The *automaton of the language L* is the deterministic automaton $\mathcal{M}_L = (A, \mathcal{Q}_L, \delta_L, L, F_L)$ is defined by $\delta_L(t^{-1}L, a) = (ta)^{-1}L$ for $t \in A^*$ and $a \in A$, and $F_L = \{x^{-1}L \mid x \in L\}$.

Remarks

- The mapping δ_L is well defined; that is, $t^{-1}L = y^{-1}L$ implies $(ta)^{-1}L = (ya)^{-1}L$. Indeed, let $w \in (ta)^{-1}L$. We have $taw \in L$ which implies $aw \in t^{-1}L = y^{-1}L$. Consequently, $yaw \in L$, so $w \in (ya)^{-1}L$. Thus, $(ta)^{-1}L \subseteq (ya)^{-1}L$. The reverse inclusion can be shown similarly, so $(ta)^{-1}L = (ya)^{-1}L$.
- We have $\delta(t^{-1}L, a) = a^{-1}(t^{-1}L)$ for every $t \in A^*$ and $a \in A$. This remark is very important for the algorithm discussed next.

We have

$$\delta_L^*(x^{-1}L, y) = (xy)^{-1}L$$

for every $x, y \in A^*$.
The argument is by induction on $\ell = |y|$. The basis case, $\ell = 0$, is immediate. Suppose that the equality holds for words of length less than ℓ, and let y be a word of length ℓ. We have $y = za$, where $z \in A^*$, $a \in A$ and $|z| = \ell - 1$. This gives:

$$\begin{aligned} & \delta_L^*(x^{-1}L, y) \\ = \ & \delta_L^*(x^{-1}L, za) = \delta_L(\delta_L^*(x^{-1}L, z), a) \\ = \ & \delta_L((xz)^{-1}L, a) = (xza)^{-1}L = (xy)^{-1}L \end{aligned}$$

The set of final states of $\mathcal{M}_L$ can now be written as

$$F_L = \{\delta^*(L, x) \mid x \in L\},$$

which allows us to compute the set F_L, once we have computed the transition function.

Nerode's Theorem:

Theorem

The language L is regular if and only if the set $\mathcal{Q}_L$ is finite.

Proof

Suppose that the set $\mathcal{Q}_L$ is finite. In this case $\mathcal{M}_L$ is a dfa and we have

$$\begin{aligned} L(\mathcal{M}_L) &= \{x \in A^* \mid \delta_L^*(L, x) \in F_L\} \\ &= \{x \in A^* \mid x^{-1}L \in F_L\}. \end{aligned}$$

From the definition of F_L it follows that $x \in L(\mathcal{M}_L)$ implies that $x^{-1}L = z^{-1}L$ for some word $z \in L$, which shows that $(x, z) \in \nu_L$. Since L is a ν_L-saturated set, this implies $x \in L$. The reverse inclusion, $L \subseteq L(\mathcal{M}_L)$ is immediate, and it is left to the reader. Therefore, L is accepted by the dfa $\mathcal{M}_L$, so L is regular.

Conversely, suppose that L is a regular language. The finiteness of the set $\mathcal{Q}_L$ follows from a previous Corollary.

Theorem

Let L be a regular language. The automaton $\mathcal{M}_L$ has the least number of states among all dfas that accept L.

Proof

Let $\mathcal{M} = (A, Q, \delta, q_0, F)$ be a dfa such that $L = L(\mathcal{M})$. We intend to show that $|\mathcal{Q}_L| \leq |Q|$. Clearly, if $\mathcal{M}$ is to be minimal, it must be accessible, otherwise the automaton resulting from removing inaccessible states accepts the same language but has fewer states. In other words, we assume that for every state $q \in Q$ there exists a word $t \in A^*$ such that $\delta^*(q_0, t) = q$.

Define the mapping $f : Q \longrightarrow \mathcal{Q}_L$ by $f(q) = t^{-1}L$ if $\delta^*(q_0, t) = q$.

Proof (cont'd)

We need to verify that f is well-defined, that is, that $\delta^*(q_0, u) = \delta^*(q_0, v)$ implies $u^{-1}L = v^{-1}L$. If $x \in u^{-1}L$, then $ux \in L$, that is, $\delta^*(q_0, ux) \in F$. Since $\delta^*(q_0, ux) = \delta^*(\delta^*(q_0, u), x)$ and $\delta^*(q_0, u) = \delta^*(q_0, v)$, it follows that $\delta^*(\delta^*(q_0, v), x) = \delta^*(q_0, vx) \in F$, so $vx \in L$ and $x \in v^{-1}L$. The reverse implication can be obtained by exchanging u and v, so f is indeed well-defined.

It is clear that the mapping f is surjective, so $|\mathcal{Q}_L| \leq |Q|$, which shows that $\mathcal{M}_L$ has the least number of states among all dfas that accept the language L.

The Algorithm

Input: A regular language L over an alphabet A.
Output: The set $\mathcal{Q}_L$ of left derivatives of L.
Method: Construct an increasing chain $\mathcal{Q}_0, \ldots, \mathcal{Q}_k, \ldots$ of finite subsets of $\mathcal{Q}_L$ as follows:

$$\begin{aligned} \mathcal{Q}_0 &= \{L\} \\ \mathcal{Q}_{k+1} &= \mathcal{Q}_k \cup \{a^{-1}K \mid a \in A \text{ and } K \in \mathcal{Q}_k\} \end{aligned}$$

Continue until $\mathcal{Q}_{k+1} = \mathcal{Q}_k$; then stop and output $\mathcal{Q}_k$.

Proof of Correctness:
The algorithm must stop, since $\mathcal{Q}_L$ is a finite set. It is easy to see that $K \in \mathcal{Q}_p$ if and only if the set K (considered as a state of the automaton $\mathcal{M}_L$) can be reached by a word of length less than or equal to p in $\mathcal{M}_L$. Every state of the automaton $\mathcal{M}_L$ can be reached through a word of length less than $|\mathcal{Q}_L|$. Therefore, when the algorithm stops, all members of $\mathcal{Q}_L$ have been computed.

We recall several equalities previously shown that are useful in the computation of left derivatives of languages. Namely, if L, K are two languages over the alphabet A and $a \in A$, then we have:

$$\begin{aligned} a^{-1}(L \cup K) &= a^{-1}L \cup a^{-1}K \\ a^{-1}(LK) &= (a^{-1}L)K \cup (L \cap \{\lambda\})a^{-1}K \\ a^{-1}L^* &= (a^{-1}L)L^* \end{aligned}$$

Example

Let $A = \{a, b\}$. Consider the regular language L that consists of all words from A^* that contain the infix aba. In other words, $L = A^*abaA^*$.

We have $\mathcal{Q}_0 = \{L\}$ and $\mathcal{Q}_1 = \mathcal{Q}_0 \cup \{a^{-1}L, b^{-1}L\}$. Note that

$$\begin{aligned} a^{-1}L &= a^{-1}(A^*abaA^*) \\ &= (a^{-1}A^*)(abaA^*) \cup (A^* \cap \{\lambda\})a^{-1}(abaA^*) \\ &= A^*abaA^* \cup baA^* \\ &= L \cup baA^* \end{aligned}$$

and

$$\begin{aligned} b^{-1}L &= b^{-1}(A^*abaA^*) \\ &= (b^{-1}A^*)(abaA^*) \cup (A^* \cap \{\lambda\})b^{-1}(abaA^*) \\ &= A^*abaA^* \\ &= L, \end{aligned}$$

because $b^{-1}(abaA^*) = \emptyset$, $(A^* \cap \{\lambda\})b^{-1}(abaA^*) = \emptyset$, and $b^{-1}A^* = A^*$.

Thus,

$$\mathcal{Q}_1 = \{L, L \cup baA^*\}.$$

Next, in order to compute $\mathcal{Q}_2$, observe that

$$\begin{aligned} a^{-1}baA^* &= \emptyset \\ b^{-1}baA^* &= aA^*. \end{aligned}$$

We obtain:

$$\begin{aligned} a^{-1}(L \cup baA^*) &= a^{-1}L = L \cup baA^* \\ b^{-1}(L \cup baA^*) &= b^{-1}L \cup aA^* = L \cup aA^*, \end{aligned}$$

To compute $\mathcal{Q}_2$, observe that

$$\begin{aligned} a^{-1}baA^* &= \emptyset \\ b^{-1}baA^* &= aA^*. \end{aligned}$$

We obtain:

$$\begin{aligned} a^{-1}(L \cup baA^*) &= a^{-1}L = L \cup baA^* \\ b^{-1}(L \cup baA^*) &= b^{-1}L \cup aA^* = L \cup aA^*, \end{aligned}$$

The collection $\mathcal{Q}_2$ is

$$\mathcal{Q}_2 = \{L, L \cup baA^*, L \cup aA^*\}$$

Now we have

$$\begin{aligned} a^{-1}aA^* &= A^* \\ b^{-1}aA^* &= \emptyset, \end{aligned}$$

which allows us to write:

$$\begin{aligned} a^{-1}(L \cup aA^*) &= a^{-1}L \cup A^* = A^* \\ b^{-1}(L \cup aA^*) &= b^{-1}L = L. \end{aligned}$$

The collection $\mathcal{Q}_3$ is given by

$$\mathcal{Q}_3 = \{L, L \cup baA^*, L \cup aA^*, A^*\}.$$

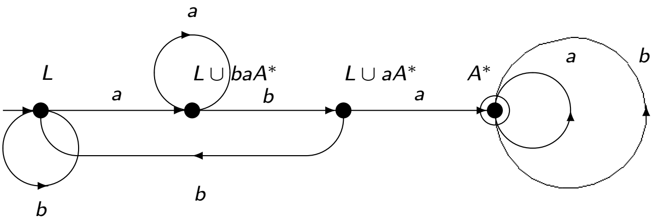
Since $a^{-1}A^* = b^{-1}A^* = A^*$, it follows that $\mathcal{Q}_4 = \mathcal{Q}_3$, so

$$\mathcal{Q}_L = \{L, L \cup baA^*, L \cup aA^*, A^*\}.$$

The automaton $\mathcal{M}_L$ is defined by the following table:

Input	State			
	L	$L \cup baA^*$	$L \cup aA^*$	A^*
a	$L \cup baA^*$	$L \cup baA^*$	A^*	A^*
b	L	$L \cup aA^*$	L	A^*

- Since $F_L = \{\delta^*(L, x) \mid x \in L\}$, we can compute F_L by determining those members of $\mathcal{Q}_L$ that can be reached from the initial state L using words from L of length not greater than 3.
- The language L contains only one word of length 3, namely *aba*, so $F_L = \{\delta^*(L, aba)\} = \{A^*\}$. The graph of $\mathcal{M}_L$ is given next.


a
L
L ∪ baA*
L ∪ aA*
A*
a
b
a
a
b
b
b