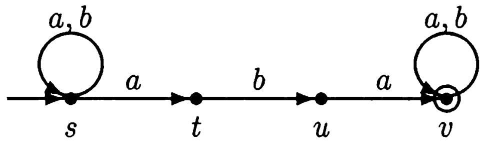
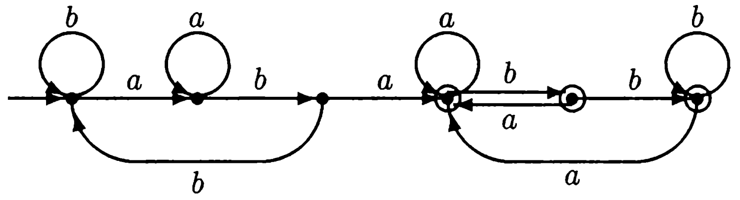
Lecture 13

DFA State Minimization

By now you have probably come across several situations in which you have observed that some automaton could be simplified either by deleting states inaccessible from the start state or by collapsing states that were equivalent in some sense. For example, if you were to apply the subset construction to the NFA

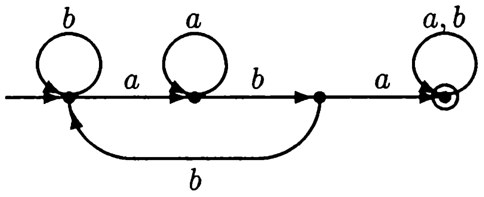


accepting the set of all strings containing the substring *aba*, you would obtain a DFA with $2^4 = 16$ states. However, all except six of these states are inaccessible. Deleting them, you would obtain the DFA



From left to right, the states of this DFA correspond to the subsets $\{s\}$, $\{s, t\}$, $\{s, u\}$, $\{s, t, v\}$, $\{s, u, v\}$, $\{s, v\}$.

Now, note that the rightmost three states of this DFA might as well be collapsed into a single state, since they are all accept states, and once the machine enters one of them it cannot escape. Thus this DFA is equivalent to



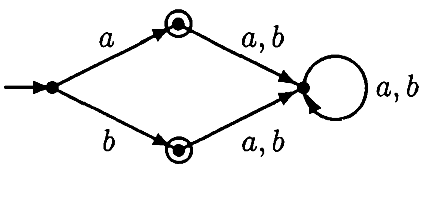
This is a simple example in which the equivalence of states is obvious, but sometimes it is not so obvious. In this and the next lecture we will develop a mechanical method to find all equivalent states of any given DFA and collapse them. This will give a DFA for any given regular set A that has as few states as possible. An amazing fact is that every regular set has a minimal DFA that is unique up to isomorphism, and there is a purely mechanical method for constructing it from any given DFA for A.

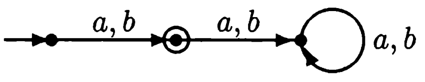
Say we are given a DFA $M = (Q,\ \Sigma,\ \delta,\ s,\ F)$ for A. The minimization process consists of two stages:

1. Get rid of inaccessible states; that is, states q for which there exists no string $x \in \Sigma^*$ such that $\widehat{\delta}(s,x) = q$.
2. Collapse "equivalent" states.

Removing inaccessible states surely does not change the set accepted. It is quite straightforward to see how to do this mechanically using depth-first search on the transition graph. Let us then assume that this has been done. For stage 2, we need to say what we mean by "equivalent" and how we do the collapsing. Let's look at some examples before giving a formal definition.

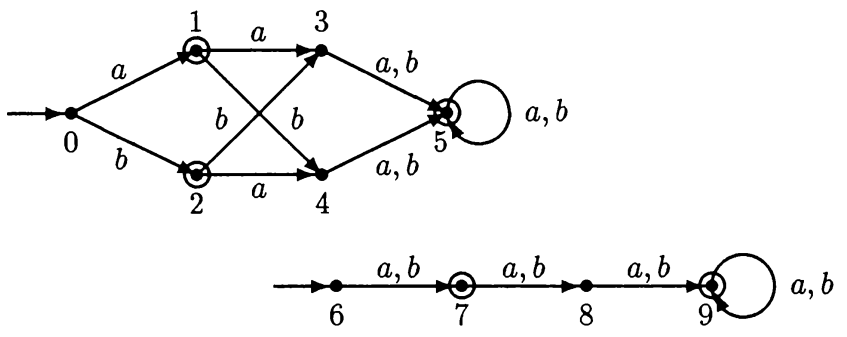
Example 13.1





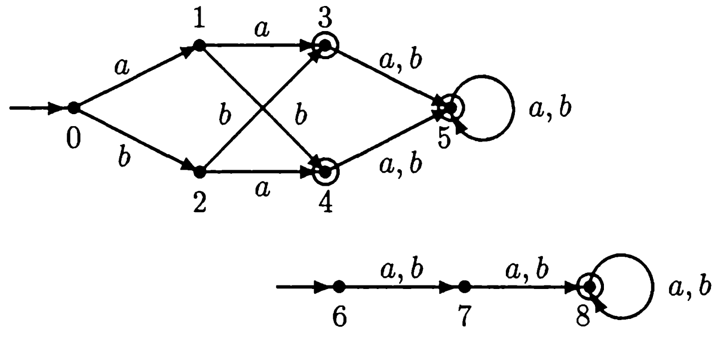
These automata both accept the set $\{a, b\}$. The automaton with four states goes to different states depending on the first input symbol, but there's really no reason for the states to be separate. They are equivalent and can be collapsed into one state, giving the automaton with three states. □

Example 13.2



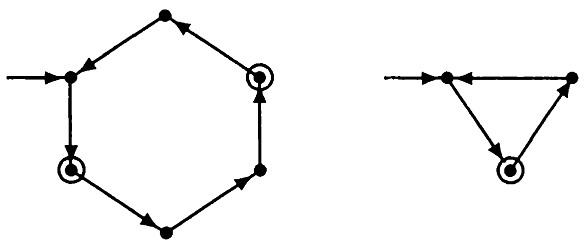
This example is a little more complicated. The automata both accept the set $\{a, b\} \cup \{\text{strings of length 3 or greater}\}$. In the first automaton, states 3 and 4 are equivalent, since they both go to state 5 under both input symbols, so there's no reason to keep them separate. Once we collapse them, we can collapse 1 and 2 for the same reason, giving the second automaton. State 0 becomes state 6; states 1 and 2 collapse to become state 7; states 3 and 4 collapse to become state 8; and state 5 becomes state 9. □

Example 13.3



Here we have modified the first automaton by making states 3, 4 accept states instead of 1, 2. Now states 3, 4, 5 are equivalent and can be collapsed. These become state 8 of the second automaton. The set accepted is the set of all strings of length at least two. □

Example 13.4



These automata both accept the set $\{a^m \mid m \equiv 1 \bmod 3\}$ (edge labels are omitted). In the left automaton, diametrically opposed states are equivalent and can be collapsed, giving the automaton on the right. □

The Quotient Construction

How do we know in general when two states can be collapsed safely without changing the set accepted? How do we do the collapsing formally? Is there a fast algorithm for doing it? How can we determine whether any further collapsing is possible?

Surely we never want to collapse an accept state p and a reject state q, because if $p = \widehat{\delta}(s, x) \in F$ and $q = \widehat{\delta}(s, y) \notin F$, then x must be accepted and y must be rejected even after collapsing, so there is no way to declare the collapsed state to be an accept or reject state without error. Also, if we collapse p and q, then we had better also collapse $\delta(p, a)$ and $\delta(q, a)$ to maintain determinism. These two observations together imply inductively that we cannot collapse p and q if $\widehat{\delta}(p, x) \in F$ and $\widehat{\delta}(q, x) \notin F$ for some string x.

It turns out that this criterion is necessary and sufficient for deciding whether a pair of states can be collapsed. That is, if there exists a string x such that $\widehat{\delta}(p, x) \in F$ and $\widehat{\delta}(q, x) \notin F$ or vice versa, then p and q cannot be safely collapsed; and if no such x exists, then they can.

Here's how we show this formally. We first define an equivalence relation $\approx$ on Q by

$$p \approx q \overset{\text{def}}{\iff} \forall x \in \Sigma^* \ (\widehat{\delta}(p, x) \in F \iff \widehat{\delta}(q, x) \in F).$$

This definition is just a formal restatement of the collapsing criterion. It is not hard to argue that the relation $\approx$ is indeed an equivalence relation: it is

- *reflexive*: $p \approx p$ for all p;
- *symmetric*: if $p \approx q$, then $q \approx p$; and
- *transitive*: if $p \approx q$ and $q \approx r$, then $p \approx r$.

As with all equivalence relations, $\approx$ partitions the set on which it is defined into disjoint *equivalence classes*:

$$[p] \overset{\text{def}}{=} \{q \mid q \approx p\}.$$

Every element $p \in Q$ is contained in exactly one equivalence class $[p]$, and

$$p \approx q \iff [p] = [q].$$

We now define a DFA $M/\approx$ called the *quotient automaton*, whose states correspond to the equivalence classes of $\approx$. This construction is called a *quotient construction* and is quite common in algebra. We will see a more general account of it in Supplementary Lectures C and D.

There is one state of $M/\approx$ for each $\approx$-equivalence class. In fact, formally, the states of $M/\approx$ *are* the equivalence classes; this is the mathematical way of "collapsing" equivalent states.

Define

$$M/\approx \stackrel{\text{def}}{=} (Q',\ \Sigma,\ \delta',\ s',\ F'),$$

where

$$\begin{aligned} Q' &\stackrel{\text{def}}{=} \{[p] \mid p \in Q\}, \\ \delta'([p],a) &\stackrel{\text{def}}{=} [\delta(p,a)], \\ s' &\stackrel{\text{def}}{=} [s], \\ F' &\stackrel{\text{def}}{=} \{[p] \mid p \in F\}. \end{aligned} \tag{13.1}$$

There is a subtle but important point involving the definition of δ' in (13.1): we need to show that it is *well-defined.* Note that the action of δ' on the equivalence class $[p]$ is defined in terms of p. It is conceivable that a different choice of representative of the class $[p]$ (i.e., some q such that $q \approx p$) might lead to a different right-hand side in (13.1). Lemma 13.5 says exactly that this does not happen.

Lemma 13.5 *If $p \approx q$, then $\delta(p,a) \approx \delta(q,a)$. Equivalently, if $[p] = [q]$, then $[\delta(p,a)] = [\delta(q,a)]$.*

Proof. Suppose $p \approx q$. Let $a \in \Sigma$ and $y \in \Sigma^*$.

$$\begin{aligned} \widehat{\delta}(\delta(p,a),y) \in F &\iff \widehat{\delta}(p,ay) \in F \\ &\iff \widehat{\delta}(q,ay) \in F \qquad \text{since } p \approx q \\ &\iff \widehat{\delta}(\delta(q,a),y) \in F. \end{aligned}$$

Since y was arbitrary, $\delta(p,a) \approx \delta(q,a)$ by definition of $\approx$. □

Lemma 13.6 *$p \in F \iff [p] \in F'$.*

Proof. The direction $\Rightarrow$ is immediate from the definition of F'. For the direction $\Leftarrow$, we need to show that if $p \approx q$ and $p \in F$, then $q \in F$. In other words, every $\approx$-equivalence class is either a subset of F or disjoint from F. This follows immediately by taking $x = \epsilon$ in the definition of $p \approx q$. □

Lemma 13.7 *For all $x \in \Sigma^*$, $\widehat{\delta'}([p],x) = [\widehat{\delta}(p,x)]$.*

Proof. By induction on $|x|$.

Basis

For $x = \epsilon$,

$$\begin{aligned} \widehat{\delta}'([p],\epsilon) &= [p] && \text{definition of } \widehat{\delta}' \\ &= [\widehat{\delta}(p,\epsilon)] && \text{definition of } \widehat{\delta}. \end{aligned}$$

Induction step

Assume $\widehat{\delta}'([p],x) = [\widehat{\delta}(p,x)]$, and let $a \in \Sigma$.

$$\begin{aligned} \widehat{\delta}'([p],xa) &= \delta'(\widehat{\delta}'([p],x),a) && \text{definition of } \widehat{\delta}' \\ &= \delta'([\widehat{\delta}(p,x)],a) && \text{induction hypothesis} \\ &= [\delta(\widehat{\delta}(p,x),a)] && \text{definition of } \delta' \\ &= [\widehat{\delta}(p,xa)] && \text{definition of } \widehat{\delta}. \end{aligned}$$

□

Theorem 13.8 $L(M/{\approx}) = L(M)$.

Proof. For $x \in \Sigma^*$,

$$\begin{aligned} x \in L(M/\approx) &\iff \widehat{\delta}'(s',x) \in F' && \text{definition of acceptance} \\ &\iff \widehat{\delta}'([s],x) \in F' && \text{definition of } s' \\ &\iff [\widehat{\delta}(s,x)] \in F' && \text{Lemma 13.7} \\ &\iff \widehat{\delta}(s,x) \in F && \text{Lemma 13.6} \\ &\iff x \in L(M) && \text{definition of acceptance.} \end{aligned}$$

□

$M/{\approx}$ Cannot Be Collapsed Further

It is conceivable that after doing the quotient construction once, we might be able to collapse even further by doing it again. It turns out that once is enough. To see this, let's do the quotient construction a second time. Define

$$[p] \sim [q] \overset{\text{def}}{\iff} \forall x \in \Sigma^* \ (\widehat{\delta}'([p],x) \in F' \iff \widehat{\delta}'([q],x) \in F').$$

This is exactly the same definition as $\approx$ above, only applied to the quotient automaton $M/{\approx}$. We use the notation $\sim$ for the equivalence relation on Q' to distinguish it from the relation $\approx$ on Q. Now

$$\begin{aligned} &[p] \sim [q] \\ &\Rightarrow \forall x \ (\widehat{\delta}'([p],x) \in F' \iff \widehat{\delta}'([q],x) \in F') && \text{definition of } \sim \\ &\Rightarrow \forall x \ ([\widehat{\delta}(p,x)] \in F' \iff [\widehat{\delta}(q,x)] \in F') && \text{Lemma 13.7} \end{aligned}$$

$\Rightarrow \forall x\ (\widehat{\delta}(p,x) \in F \iff \widehat{\delta}(q,x) \in F)$	Lemma 13.6
$\Rightarrow p \approx q$	definition of $\approx$
$\Rightarrow [p] = [q]$.	

Thus any two equivalent states of $M/\approx$ are in fact equal, and the collapsing relation $\sim$ on Q' is just the identity relation $=$.

Lecture 14

A Minimization Algorithm

Here is an algorithm for computing the collapsing relation $\approx$ for a given DFA M with no inaccessible states. Our algorithm will mark (unordered) pairs of states $\{p, q\}$. A pair $\{p, q\}$ will be marked as soon as a reason is discovered why p and q are *not* equivalent.

1. Write down a table of all pairs $\{p, q\}$, initially unmarked.
2. Mark $\{p, q\}$ if $p \in F$ and $q \notin F$ or vice versa.
3. Repeat the following until no more changes occur: if there exists an unmarked pair $\{p, q\}$ such that $\{\delta(p, a), \delta(q, a)\}$ is marked for some $a \in \Sigma$, then mark $\{p, q\}$.
4. When done, $p \approx q$ iff $\{p, q\}$ is not marked.

Here are some things to note about this algorithm:

- If $\{p, q\}$ is marked in step 2, then p and q are surely not equivalent: take $x = \epsilon$ in the definition of $\approx$.
- We may have to look at the same pair $\{p, q\}$ many times in step 3, since any change in the table may suddenly allow $\{p, q\}$ to be marked. We stop only after we make an entire pass through the table with no new marks.

- The algorithm runs for only a finite number of steps, since there are only $\binom{n}{2}$ possible marks that can be made,[1] and we have to make at least one new mark in each pass to keep going.
- Step 4 is really a statement of the theorem that the algorithm correctly computes $\approx$. This requires proof, which we defer until later.

Example 14.1 Let's minimize the automaton of Example 13.2 of Lecture 13.

		a	b
$\rightarrow$	0	1	2
	$1F$	3	4
	$2F$	4	3
	3	5	5
	4	5	5
	$5F$	5	5

Here is the table built in step 1. Initially all pairs are unmarked.

0					
–	1				
–	–	2			
–	–	–	3		
–	–	–	–	4	
–	–	–	–	–	5

After step 2, all pairs consisting of one accept state and one nonaccept state have been marked.

0					
✓	1				
✓	–	2			
–	✓	✓	3		
–	✓	✓	–	4	
✓	–	–	✓	✓	5

Now look at an unmarked pair, say $\{0,3\}$. Under input a, 0 and 3 go to 1 and 5, respectively (write: $\{0,3\} \rightarrow \{1,5\}$). The pair $\{1,5\}$ is not marked, so we don't mark $\{0,3\}$, at least not yet. Under input b, $\{0,3\} \rightarrow \{2,5\}$, which is not marked, so we still don't mark $\{0,3\}$. We then look at unmarked pairs $\{0,4\}$ and $\{1,2\}$ and find out we cannot mark them yet for the same reasons. But for $\{1,5\}$, under input a, $\{1,5\} \rightarrow \{3,5\}$, and $\{3,5\}$ is marked, so we mark $\{1,5\}$. Similarly, under input a, $\{2,5\} \rightarrow \{4,5\}$ which is marked, so we mark $\{2,5\}$. Under both inputs a and b, $\{3,4\} \rightarrow \{5,5\}$, which is never marked (it's not even in the table), so we do not mark $\{3,4\}$. After the first

[1] $\binom{n}{k} \overset{\text{def}}{=} \frac{n!}{k!(n-k)!}$, the number of subsets of size k in a set of size n.

pass of step 3, the table looks like

0					
✓	1				
✓	–	2			
–	✓	✓	3		
–	✓	✓	–	4	
✓	✓	✓	✓	✓	5

Now we make another pass through the table. As before, $\{0,3\} \to \{1,5\}$ under input a, but this time $\{1,5\}$ is marked, so we mark $\{0,3\}$. Similarly, $\{0,4\} \to \{2,5\}$ under input b, and $\{2,5\}$ is marked, so we mark $\{0,4\}$. This gives

0					
✓	1				
✓	–	2			
✓	✓	✓	3		
✓	✓	✓	–	4	
✓	✓	✓	✓	✓	5

Now we check the remaining unmarked pairs and find out that $\{1,2\} \to \{3,4\}$ and $\{3,4\} \to \{5,5\}$ under both a and b, and neither $\{3,4\}$ nor $\{5,5\}$ is marked, so there are no new marks. We are left with unmarked pairs $\{1,2\}$ and $\{3,4\}$, indicating that $1 \approx 2$ and $3 \approx 4$. □

Example 14.2 Now let's do Example 13.4 of Lecture 13.

		a
→	0	1
	1F	2
	2	3
	3	4
	4F	5
	5	0

Here is the table after step 2.

0					
✓	1				
–	✓	2			
–	✓	–	3		
✓	–	✓	✓	4	
–	✓	–	–	✓	5

Then:

- $\{0,2\} \to \{1,3\}$, which is marked, so mark $\{0,2\}$.

- $\{0,3\} \rightarrow \{1,4\}$, which is not marked, so do not mark $\{0,3\}$.
- $\{0,5\} \rightarrow \{0,1\}$, which is marked, so mark $\{0,5\}$.
- $\{1,4\} \rightarrow \{2,5\}$, which is not marked, so do not mark $\{1,4\}$.
- $\{2,3\} \rightarrow \{3,4\}$, which is marked, so mark $\{2,3\}$.
- $\{2,5\} \rightarrow \{0,3\}$, which is not marked, so do not mark $\{2,5\}$.
- $\{3,5\} \rightarrow \{0,4\}$, which is marked, so mark $\{3,5\}$.

After the first pass, the table looks like this:

0					
✓	1				
✓	✓	2			
–	✓	✓	3		
✓	–	✓	✓	4	
✓	✓	–	✓	✓	5

Now do another pass. We discover that $\{0,3\} \rightarrow \{1,4\} \rightarrow \{2,5\} \rightarrow \{0,3\}$ and none of these are marked, so we are done. Thus $0 \approx 3$, $1 \approx 4$, and $2 \approx 5$. □

Correctness of the Collapsing Algorithm

Theorem 14.3 *The pair $\{p,q\}$ is marked by the above algorithm if and only if there exists $x \in \Sigma^*$ such that $\widehat{\delta}(p,x) \in F$ and $\widehat{\delta}(q,x) \notin F$ or vice versa; i.e., if and only if $p \not\approx q$.*

Proof. This is easily proved by induction. We leave the proof as an exercise (Miscellaneous Exercise 49). □

A nice way to look at the algorithm is as a finite automaton itself. Let

$$\mathcal{Q} = \{\{p,q\} \mid p,q \in Q,\ p \neq q\}.$$

There are $\binom{n}{2}$ elements of $\mathcal{Q}$, where n is the size of Q. Define a nondeterministic "transition function"

$$\Delta : \mathcal{Q} \rightarrow 2^{\mathcal{Q}}$$

on $\mathcal{Q}$ as follows:

$$\Delta(\{p,q\},a) = \{\{p',q'\} \mid p = \delta(p',a),\ q = \delta(q',a)\}.$$

Define a set of "start states" $\mathcal{S} \subseteq \mathcal{Q}$ as follows:

$$\mathcal{S} = \{\{p,q\} \mid p \in F,\ q \notin F\}.$$

(We don't need to write "...or vice versa" because $\{p, q\}$ is an unordered pair.) Step 2 of the algorithm marks the elements of $\mathcal{S}$, and step 3 marks pairs in $\Delta(\{p, q\}, a)$ when $\{p, q\}$ is marked for any $a \in \Sigma$. In these terms, Theorem 14.3 says that $p \not\approx q$ iff $\{p, q\}$ is accessible in this automaton.

Lecture 15

Myhill–Nerode Relations

Two deterministic finite automata

$$M = (Q_M,\ \Sigma,\ \delta_M,\ s_M,\ F_M),$$
$$N = (Q_N,\ \Sigma,\ \delta_N,\ s_N,\ F_N)$$

are said to be *isomorphic* (Greek for "same form") if there is a one-to-one and onto mapping $f : Q_M \to Q_N$ such that

- $f(s_M) = s_N$,
- $f(\delta_M(p, a)) = \delta_N(f(p), a)$ for all $p \in Q_M$, $a \in \Sigma$, and
- $p \in F_M$ iff $f(p) \in F_N$.

That is, they are essentially the same automaton up to renaming of states. It is easily argued that isomorphic automata accept the same set.

In this lecture and the next we will show that if M and N are any two automata with no inaccessible states accepting the same set, then the quotient automata $M/{\approx}$ and $N/{\approx}$ obtained by the collapsing algorithm of Lecture 14 are isomorphic. Thus the DFA obtained by the collapsing algorithm is the minimal DFA for the set it accepts, and this automaton is unique up to isomorphism.

We will do this by exploiting a profound and beautiful correspondence between finite automata with input alphabet Σ and certain equivalence relations on Σ^*. We will show that the unique minimal DFA for a regular set R can be defined in a natural way *directly from* R, and that any minimal automaton for R is isomorphic to this automaton.

Myhill–Nerode Relations

Let $R \subseteq \Sigma^*$ be a regular set, and let $M = (Q, \Sigma, \delta, s, F)$ be a DFA for R with no inaccessible states. The automaton M induces an equivalence relation $\equiv_M$ on Σ^* defined by

$$x \equiv_M y \overset{\text{def}}{\iff} \widehat{\delta}(s,x) = \widehat{\delta}(s,y).$$

(Don't confuse this relation with the collapsing relation $\approx$ of Lecture 13—that relation was defined on Q, whereas $\equiv_M$ is defined on Σ^*.)

One can easily show that the relation $\equiv_M$ is an equivalence relation; that is, that it is reflexive, symmetric, and transitive. In addition, $\equiv_M$ satisfies a few other useful properties:

(i) It is a *right congruence*: for any $x, y \in \Sigma^*$ and $a \in \Sigma$,

$$x \equiv_M y \Rightarrow xa \equiv_M ya.$$

To see this, assume that $x \equiv_M y$. Then

$$\begin{aligned}\widehat{\delta}(s,xa) &= \delta(\widehat{\delta}(s,x),a)\\ &= \delta(\widehat{\delta}(s,y),a) \qquad \text{by assumption}\\ &= \widehat{\delta}(s,ya).\end{aligned}$$

(ii) It *refines* R: for any $x, y \in \Sigma^*$,

$$x \equiv_M y \Rightarrow (x \in R \iff y \in R).$$

This is because $\widehat{\delta}(s,x) = \widehat{\delta}(s,y)$, and this is either an accept or a reject state, so either both x and y are accepted or both are rejected. Another way to say this is that every $\equiv_M$-class has either all its elements in R or none of its elements in R; in other words, R is a union of $\equiv_M$-classes.

(iii) It is of *finite index*; that is, it has only finitely many equivalence classes. This is because there is exactly one equivalence class

$$\{x \in \Sigma^* \mid \widehat{\delta}(s,x) = q\}$$

corresponding to each state q of M.

Let us call an equivalence relation $\equiv$ on Σ^* a *Myhill–Nerode relation for* R if it satisfies properties (i), (ii), and (iii); that is, if it is a right congruence of finite index refining R.

The interesting thing about this definition is that it characterizes exactly the relations on Σ^* that are $\equiv_M$ for some automaton M. In other words, we can reconstruct M from $\equiv_M$ using only the fact that $\equiv_M$ is Myhill–Nerode. To see this, we will show how to construct an automaton $M_\equiv$ for R from any given Myhill–Nerode relation $\equiv$ for R. We will show later that the two constructions

$$\begin{aligned} &M \mapsto \equiv_M, \\ &\equiv \mapsto M_\equiv \end{aligned}$$

are inverses up to isomorphism of automata.

Let $R \subseteq \Sigma^*$, and let $\equiv$ be an arbitrary Myhill–Nerode relation for R. Right now we're not assuming that R is regular, only that the relation $\equiv$ satisfies (i), (ii), and (iii). The $\equiv$-class of the string x is

$$[x] \stackrel{\text{def}}{=} \{y \mid y \equiv x\}.$$

Although there are infinitely many strings, there are only finitely many $\equiv$-classes, by property (iii).

Now define the DFA $M_\equiv = (Q,\ \Sigma,\ \delta,\ s,\ F)$, where

$$\begin{aligned} Q &\stackrel{\text{def}}{=} \{[x] \mid x \in \Sigma^*\}, \\ s &\stackrel{\text{def}}{=} [\epsilon], \\ F &\stackrel{\text{def}}{=} \{[x] \mid x \in R\}, \\ \delta([x], a) &\stackrel{\text{def}}{=} [xa]. \end{aligned}$$

It follows from property (i) of Myhill–Nerode relations that δ is well defined. In other words, we have defined the action of δ on an equivalence class $[x]$ in terms of an element x chosen from that class, and it is conceivable that we could have gotten something different had we chosen another $y \in [x]$ such that $[xa] \neq [ya]$. The property of right congruence says exactly that this cannot happen.

Finally, observe that

$$x \in R \iff [x] \in F. \tag{15.1}$$

The implication $(\Rightarrow)$ is from the definition of F, and $(\Leftarrow)$ follows from the definition of F and property (ii) of Myhill–Nerode relations.

Now we are ready to prove that $L(M_\equiv) = R$.

Lemma 15.1 $\widehat{\delta}([x], y) = [xy]$.

Proof. Induction on $|y|$.

Basis

$$\widehat{\delta}([x],\epsilon) = [x] = [x\epsilon].$$

Induction step

$$\begin{aligned}\widehat{\delta}([x],ya) &= \delta(\widehat{\delta}([x],y),a) && \text{definition of } \widehat{\delta}\\ &= \delta([xy],a) && \text{induction hypothesis}\\ &= [xya] && \text{definition of } \delta.\end{aligned}$$ □

Theorem 15.2 $L(M_\equiv) = R$.

Proof.

$$\begin{aligned}x \in L(M_\equiv) &\iff \widehat{\delta}([\epsilon],x) \in F && \text{definition of acceptance}\\ &\iff [x] \in F && \text{Lemma 15.1}\\ &\iff x \in R && \text{property (15.1).}\end{aligned}$$ □

$M \mapsto\ \equiv_M$ and $\equiv\ \mapsto M_\equiv$ Are Inverses

We have described two natural constructions, one taking a given automaton M for R with no inaccessible states to a corresponding Myhill–Nerode relation $\equiv_M$ for R, and one taking a given Myhill–Nerode relation $\equiv$ for R to a DFA $M_\equiv$ for R. We now wish to show that these two operations are inverses up to isomorphism.

Lemma 15.3 *(i) If $\equiv$ is a Myhill–Nerode relation for R, and if we apply the construction $\equiv\ \mapsto M_\equiv$ and then apply the construction $M \mapsto\ \equiv_M$ to the result, the resulting relation $\equiv_{M_\equiv}$ is identical to $\equiv$.*

(ii) If M is a DFA for R with no inaccessible states, and if we apply the construction $M \mapsto\ \equiv_M$ and then apply the construction $\equiv\ \mapsto M_\equiv$ to the result, the resulting DFA $M_{\equiv_M}$ is isomorphic to M.

Proof. (i) Let $M_\equiv = (Q,\ \Sigma,\ \delta,\ s,\ F)$ be the automaton constructed from $\equiv$ as described above. Then for any $x, y \in \Sigma^*$,

$$\begin{aligned}x \equiv_{M_\equiv} y &\iff \widehat{\delta}(s,x) = \widehat{\delta}(s,y) && \text{definition of } \equiv_{M_\equiv}\\ &\iff \widehat{\delta}([\epsilon],x) = \widehat{\delta}([\epsilon],y) && \text{definition of } s\\ &\iff [x] = [y] && \text{Lemma 15.1}\\ &\iff x \equiv y.\end{aligned}$$

(ii) Let $M = (Q, \Sigma, \delta, s, F)$ and let $M_{\equiv_M} = (Q', \Sigma, \delta', s', F')$. Recall from the construction that

$$\begin{aligned}
[x] &= \{y \mid y \equiv_M x\} = \{y \mid \widehat{\delta}(s,y) = \widehat{\delta}(s,x)\}, \\
Q' &= \{[x] \mid x \in \Sigma^*\}, \\
s' &= [\epsilon], \\
F' &= \{[x] \mid x \in R\}, \\
\delta'([x],a) &= [xa].
\end{aligned}$$

We will show that $M_{\equiv_M}$ and M are isomorphic under the map

$$\begin{aligned}
&f : Q' \to Q, \\
&f([x]) = \widehat{\delta}(s,x).
\end{aligned}$$

By the definition of $\equiv_M$, $[x] = [y]$ iff $\widehat{\delta}(s,x) = \widehat{\delta}(s,y)$, so the map f is well defined on $\equiv_M$-classes and is one-to-one. Since M has no inaccessible states, f is onto.

To show that f is an isomorphism of automata, we need to show that f preserves all automata-theoretic structure: the start state, transition function, and final states. That is, we need to show

- $f(s') = s$,
- $f(\delta'([x],a)) = \delta(f([x]),a)$,
- $[x] \in F' \iff f([x]) \in F$.

These are argued as follows:

$$\begin{aligned}
f(s') &= f([\epsilon]) && \text{definition of } s' \\
&= \widehat{\delta}(s,\epsilon) && \text{definition of } f \\
&= s && \text{definition of } \widehat{\delta};
\end{aligned}$$

$$\begin{aligned}
f(\delta'([x],a)) &= f([xa]) && \text{definition of } \delta' \\
&= \widehat{\delta}(s,xa) && \text{definition of } f \\
&= \delta(\widehat{\delta}(s,x),a) && \text{definition of } \widehat{\delta} \\
&= \delta(f([x]),a) && \text{definition of } f;
\end{aligned}$$

$$\begin{aligned}
[x] \in F' &\iff x \in R && \text{definition of } F \text{ and property (ii)} \\
&\iff \widehat{\delta}(s,x) \in F && \text{since } L(M) = R \\
&\iff f([x]) \in F && \text{definition of } f.
\end{aligned}$$

□

We have shown:

Theorem 15.4 *Let Σ be a finite alphabet. Up to isomorphism of automata, there is a one-to-one correspondence between deterministic finite automata over Σ with no inaccessible states accepting R and Myhill–Nerode relations for R on Σ^*.*

Lecture 16

The Myhill–Nerode Theorem

Let $R \subseteq \Sigma^*$ be a regular set. Recall from Lecture 15 that a *Myhill–Nerode relation for R* is an equivalence relation $\equiv$ on Σ^* satisfying the following three properties:

(i) $\equiv$ is a *right congruence*: for any $x, y \in \Sigma^*$ and $a \in \Sigma$,

$$x \equiv y \Rightarrow xa \equiv ya;$$

(ii) $\equiv$ *refines* R: for any $x, y \in \Sigma^*$,

$$x \equiv y \Rightarrow (x \in R \Longleftrightarrow y \in R);$$

(iii) $\equiv$ is of *finite index*; that is, $\equiv$ has only finitely many equivalence classes.

We showed that there was a natural one-to-one correspondence (up to isomorphism of automata) between

- deterministic finite automata for R with input alphabet Σ and with no inaccessible states, and
- Myhill–Nerode relations for R on Σ^*.

This is interesting, because it says we can deal with regular sets and finite automata in terms of a few simple, purely algebraic properties.

In this lecture we will show that there exists a *coarsest* Myhill–Nerode relation $\equiv_R$ for any given regular set R; that is, one that every other Myhill–Nerode relation for R refines. The notions of *coarsest* and *refinement* will be defined below. The relation $\equiv_R$ corresponds to the unique minimal DFA for R.

Recall from Lecture 15 the two constructions

- $M \mapsto \equiv_M$, which takes an arbitrary DFA $M = (Q, \Sigma, \delta, s, F)$ with no inaccessible states accepting R and produces a Myhill–Nerode relation $\equiv_M$ for R:

$$x \equiv_M y \overset{\text{def}}{\iff} \widehat{\delta}(s,x) = \widehat{\delta}(s,y);$$

- $\equiv \mapsto M_\equiv$, which takes an arbitrary Myhill–Nerode relation $\equiv$ on Σ^* for R and produces a DFA $M_\equiv = (Q, \Sigma, \delta, s, F)$ accepting R:

$$\begin{aligned}
[x] &\overset{\text{def}}{=} \{y \mid y \equiv x\},\\
Q &\overset{\text{def}}{=} \{[x] \mid x \in \Sigma^*\},\\
s &\overset{\text{def}}{=} [\epsilon],\\
\delta([x],a) &\overset{\text{def}}{=} [xa],\\
F &\overset{\text{def}}{=} \{[x] \mid x \in R\}.
\end{aligned}$$

We showed that these two constructions are inverses up to isomorphism.

Definition 16.1 A relation $\equiv_1$ is said to *refine* another relation $\equiv_2$ if $\equiv_1 \subseteq \equiv_2$, considered as sets of ordered pairs. In other words, $\equiv_1$ *refines* $\equiv_2$ if for all x and y, $x \equiv_1 y$ implies $x \equiv_2 y$. For equivalence relations $\equiv_1$ and $\equiv_2$, this is the same as saying that for every x, the $\equiv_1$-class of x is included in the $\equiv_2$-class of x. □

For example, the equivalence relation $x \equiv y \bmod 6$ on the integers refines the equivalence relation $x \equiv y \bmod 3$. For another example, clause (ii) of the definition of Myhill–Nerode relations says that a Myhill–Nerode relation $\equiv$ for R refines the equivalence relation with equivalence classes R and $\Sigma^* - R$.

The relation of *refinement* between equivalence relations is a partial order: it is reflexive (every relation refines itself), transitive (if $\equiv_1$ refines $\equiv_2$ and $\equiv_2$ refines $\equiv_3$, then $\equiv_1$ refines $\equiv_3$), and antisymmetric (if $\equiv_1$ refines $\equiv_2$ and $\equiv_2$ refines $\equiv_1$, then $\equiv_1$ and $\equiv_2$ are the same relation).

If $\equiv_1$ refines $\equiv_2$, then $\equiv_1$ is the *finer* and $\equiv_2$ is the *coarser* of the two relations. There is always a finest and a coarsest equivalence relation on any set U, namely the *identity relation* $\{(x,x) \mid x \in U\}$ and the *universal relation* $\{(x,y) \mid x,y \in U\}$, respectively.

Now let $R \subseteq \Sigma^*$, regular or not. We define an equivalence relation $\equiv_R$ on Σ^* in terms of R as follows:

$$x \equiv_R y \overset{\text{def}}{\iff} \forall z \in \Sigma^* \; (xz \in R \iff yz \in R). \tag{16.1}$$

In other words, two strings are equivalent under $\equiv_R$ if, whenever you append the same string to both of them, the resulting two strings are either both in R or both not in R. It is not hard to show that this is an equivalence relation for any R.

We show that for any set R, regular or not, the relation $\equiv_R$ satisfies the first two properties (i) and (ii) of Myhill–Nerode relations and is the coarsest such relation on Σ^*. In case R is regular, this relation is also of finite index, therefore a Myhill–Nerode relation for R. In fact, it is the coarsest possible Myhill–Nerode relation for R and corresponds to the unique minimal finite automaton for R.

Lemma 16.2 *Let $R \subseteq \Sigma^*$, regular or not. The relation $\equiv_R$ defined by (16.1) is a right congruence refining R and is the coarsest such relation on Σ^*.*

Proof. To show that $\equiv_R$ is a right congruence, take $z = aw$ in the definition of $\equiv_R$:

$$\begin{aligned} x \equiv_R y &\Rightarrow \forall a \in \Sigma \; \forall w \in \Sigma^* (xaw \in R \iff yaw \in R) \\ &\Rightarrow \forall a \in \Sigma \; (xa \equiv_R ya). \end{aligned}$$

To show that $\equiv_R$ refines R, take $z = \epsilon$ in the definition of $\equiv_R$:

$$x \equiv_R y \Rightarrow (x \in R \iff y \in R).$$

Moreover, $\equiv_R$ is the coarsest such relation, because any other equivalence relation $\equiv$ satisfying (i) and (ii) refines $\equiv_R$:

$$\begin{aligned} & x \equiv y \\ &\Rightarrow \forall z \; (xz \equiv yz) && \text{by induction on } |z|\text{, using property (i)} \\ &\Rightarrow \forall z \; (xz \in R \iff yz \in R) && \text{property (ii)} \\ &\Rightarrow x \equiv_R y && \text{definition of } \equiv_R. \end{aligned}$$

□

At this point all the hard work is done. We can now state and prove the *Myhill–Nerode theorem*:

Theorem 16.3 **(Myhill–Nerode theorem)** *Let $R \subseteq \Sigma^*$. The following statements are equivalent:*

(a) R is regular;

(b) there exists a Myhill–Nerode relation for R;

(c) the relation $\equiv_R$ is of finite index.

Proof. (a) $\Rightarrow$ (b) Given a DFA M for R, the construction $M \mapsto \equiv_M$ produces a Myhill–Nerode relation for R.

(b) $\Rightarrow$ (c) By Lemma 16.2, any Myhill–Nerode relation for R is of finite index and refines $\equiv_R$; therefore $\equiv_R$ is of finite index.

(c) $\Rightarrow$ (a) If $\equiv_R$ is of finite index, then it is a Myhill–Nerode relation for R, and the construction $\equiv \mapsto M_\equiv$ produces a DFA for R. □

Since $\equiv_R$ is the unique coarsest Myhill–Nerode relation for a regular set R, it corresponds to the DFA for R with the fewest states among all DFAs for R.

The collapsing algorithm of Lecture 14 actually gives this automaton. Suppose $M = (Q, \Sigma, \delta, s, F)$ is a DFA for R that is already collapsed; that is, there are no inaccessible states, and the collapsing relation

$$p \approx q \overset{\text{def}}{\iff} \forall x \in \Sigma^* \ (\widehat{\delta}(p,x) \in F \iff \widehat{\delta}(q,x) \in F)$$

is the identity relation on Q. Then the Myhill–Nerode relation $\equiv_M$ corresponding to M is exactly $\equiv_R$:

$$\begin{array}{ll}
x \equiv_R y & \\
\iff \forall z \in \Sigma^* \ (xz \in R \iff yz \in R) & \text{definition of } \equiv_R \\
\iff \forall z \in \Sigma^* \ (\widehat{\delta}(s,xz) \in F \iff \widehat{\delta}(s,yz) \in F) & \text{definition of acceptance} \\
\iff \forall z \in \Sigma^* \ (\widehat{\delta}(\widehat{\delta}(s,x),z) \in F \iff \widehat{\delta}(\widehat{\delta}(s,y),z) \in F) & \\
 & \text{Homework 1, Exercise 3} \\
\iff \widehat{\delta}(s,x) \approx \widehat{\delta}(s,y) & \text{definition of } \approx \\
\iff \widehat{\delta}(s,x) = \widehat{\delta}(s,y) & \text{since } M \text{ is collapsed} \\
\iff x \equiv_M y & \text{definition of } \equiv_M.
\end{array}$$

An Application

The Myhill–Nerode theorem can be used to determine whether a set R is regular or nonregular by determining the number of $\equiv_R$-classes. For example, consider the set

$$A = \{a^n b^n \mid n \geq 0\}.$$

If $k \neq m$, then $a^k \not\equiv_A a^m$, since $a^k b^k \in A$ but $a^m b^k \notin A$. Therefore, there are infinitely many $\equiv_A$-classes, at least one for each a^k, $k \geq 0$. By the Myhill–Nerode theorem, A is not regular.

In fact, one can show that the $\equiv_A$-classes are exactly

$$G_k = \{a^k\}, \quad k \geq 0,$$

$$H_k = \{a^{n+k}b^n \mid 1 \leq n\}, \quad k \geq 0,$$
$$E = \Sigma^* - \bigcup_{k\geq 0} G_k \cup H_k = \Sigma^* - \{a^m b^n \mid 0 \leq n \leq m\}.$$

For strings in G_k, all and only strings in $\{a^n b^{n+k} \mid n \geq 0\}$ can be appended to obtain a string in A; for strings in H_k, only the string b^k can be appended to obtain a string in A; and no string can be appended to a string in E to obtain a string in A.

We will see another application of the Myhill–Nerode theorem involving two-way finite automata in Lectures 17 and 18.

Historical Notes

Minimization of DFAs was studied by Huffman [61], Moore [90], Nerode [94], and Hopcroft [59], among others. The Myhill–Nerode theorem is due independently to Myhill [91] and Nerode [94] in slightly different forms.