Notes on the Myhill-Nerode Theorem

These notes present a technique to prove a lower bound on the number of states of any DFA that recognizes a given language. The technique can also be used to prove that a language is not regular. (By showing that *for every* k one needs at least k states to recognize the language.)

1 Distinguishable and Indistinguishable States

It will be helpful to keep in mind the following two languages over the alphabet $\Sigma = \{0, 1\}$ as examples: the language $EQ = \{0^n 1^n | n \geq 1\}$ of strings containing a sequence of zeroes followed by an equally long sequence of ones, and the language $A = (0 \cup 1)^* \cdot 1 \cdot (0 \cup 1)$ of strings containing a 1 in the second-to-last position.

We start with the following basic notion.

Definition 1 (Distinguishable Strings) *Let L be a language over an alphabet Σ. We say that two strings x and y are* **distinguishable** *with respect to L if there is a string z such that $xz \in L$ and $yz \notin L$, or vice versa.*

For example the strings $x = 0$ and $y = 00$ are distinguishable with respect to EQ, because if we take $z = 1$ we have $xz = 01 \in EQ$ and $yz = 001 \notin EQ$. Also, the strings $x = 00$ and y=01 are distinguishable with respect to A as can be seen by taking $z = 0$.

On the other hand, the strings $x = 0110$ and $y = 10$ are *not* distinguishable with respect to EQ because for every z we have $xz \notin EQ$ and $yz \notin EQ$.

Exercise 1 *Find two strings that are not distinguishable with respect to A.*

The intuition behind Definition 1 is captured by the following simple fact.

Lemma 2 *Let L be a language, M be a DFA that decides L, and x and y be distinguishable strings with respect to L. Then the state reached by M on input x is different from the state reached by M on input y.*

PROOF: Suppose by contradiction that M reaches the same state q on input x and on input y. Let z be the string such that $xz \in L$ and $yz \notin L$ (or vice versa). Let us call q' the state reached by M on input xz. Note that q' is the state reached by M starting from q and given the string z. But also, on input yz, M must reach the same state q', because M reaches state q given y, and then goes from q to q' given z. This means that M either accepts both xz and yz, or it rejects both. In either case, M is incorrect and we reach a contradiction. $\Box$

Consider now the following generalization of the notion of distinguishability.

Definition 3 (Distinguishable Set of Strings) *Let L be a language. A set of strings $\{x_1, \ldots, x_k\}$ is distinguishable if for every two distinct strings x_i, x_j we have that x_i is distinguishable from x_j.*

For example one can verify that $\{0, 00, 000\}$ are distinguishable with respect to EQ and that $\{00, 01, 10, 11\}$ are distinguishable with respect to A.

We now prove the main result of this section.

Lemma 4 *Let L be a language, and suppose there is a set of k distinguishable strings with respect to L. Then every DFA for L has at least k states.*

PROOF: If L is not regular, then there is no DFA for L, much less a DFA with less than k states. If L is regular, let M be a DFA for L, let $x_1, \ldots, x_k$ be the distinguishable strings, and let q_i be the state reached by M after reading x_i. For every $i \neq j$, we have that x_i and x_j are distinguishable, and so $q_i \neq q_j$ because of Lemma 2. So we have k different states $q_1, \ldots, q_k$ in M, and so M has at least k states. $\Box$

Using Lemma 4 and the fact that the strings $\{00, 01, 10, 11\}$ are distinguishable with respect to A we conclude that every DFA for A has at least 4 states.

For every $k \geq 1$, consider the set $\{0, 00, \ldots, 0^k\}$ of strings made of k or fewer zeroes. It is easy to see that this is a set of distinguishable strings with respect to EQ. This means that there cannot be a DFA for EQ, because, if there were one, it would have to have at least k states for every k, which is clearly impossible.

2 The Myhill-Nerode Theorem

Let L be a language over an alphabet Σ. We have seen in the previous section the definition of distinguishable strings with respect to L. We say that two strings x and

y are **indistinguishable**, and we write it $x \approx_L y$ if they are not distinguishable. That is, $x \approx_L y$ means that, for every string z, the string xz belongs to L if and only if the string yz does. By definition, $x \approx_L y$ if and only if $y \approx_L x$, and we always have $x \approx_L x$. It is also easy to see that if $x \approx_L y$ and $y \approx_L w$ then we must have $x \approx_L w$. In other words:

Fact 5 *The relation $\approx_L$ is an* **equivalence** *relation over the strings in Σ^*.*

As you may remember from earlier lectures, when you define an equivalence relation over a set you also define a way to partition the set into a collection of subsets, called *equivalence class*. An equivalence class in Σ^* with respect to $\approx_L$ is a set of strings that are all indistinguishable from one another, and that are all distinguishable from all the others not in the set. We denote by $[x]$ the equivalence class that contains the string x.

A fancy way of stating Lemma 4 is to say that every DFA for L must have at least as many states as the number of equivalence class in Σ^* with respect to $\approx_L$. Perhaps surprisingly, the converse is also true: there is always a DFA that has *precisely* as many states as the number of equivalence classes.

Theorem 6 (Myhill-Nerode) *Let L be a language over Σ. If Σ^* has infinitely many equivalence classes with respect to $\approx_L$, then L is not regular. Otherwise, L can be decided by a DFA whose number of states is equal to the number of equivalence classes in Σ^* with respect to $\approx_L$.*

PROOF: If there are infinitely many equivalence classes, then it follows from Lemma 4 that no DFA can decide L, and so L is not regular.

Suppose then that there is a finite number of equivalence class. We define an automaton that has a state for each equivalence class. The start state is the class $[\epsilon]$ and every state of the form $[x]$ for $x \in L$ is a final state.

It remains to describe the transition function. From a state $[x]$, reading the character a, the automaton moves to state $[xa]$. We need to make sure that this definition makes sense. If $x \approx_L x'$, then the state $[x]$ and the state $[x']$ are the same, so we need to verify that the state $[xa]$ and the state $[x'a]$ are also the same. That is, we need to verify that, for every string z, $xaz \in L$ if and only if $x'az \in L$; this is clearly true because x and x' are indistinguishable and so appending the string az makes xaz an element of the language L if and only if it also makes $x'az$ an element of the language.

So the automaton is well defined. Let now $x = x_1 x_2 \cdots x_n$ be an input string for the automaton. The automaton starts in $[\epsilon]$, then moves to state $[x_1]$, and so on, and at the end is in state $[x_1 \cdots x_n]$; this is an accepting state if and only if $x \in L$, and so the automaton works correctly on x. $\square$