IITM-CS2200: Languages, Machines and Computation Jan 25, 2012

Lecture 12 : Myhill-Nerode Relations

Lecturer: Jayalal Sarma *Scribe: Jayalal Sarma*

Over past two lectures, we developed a strategy for proving that some languages are not regular. As we noticed, proving such impossibility results is a difficult task intuitively because one has to rule out all "smart designs" possible for a finite automata. The strategy was to observe some structural property of the automata. So far, the structural study was simply making use of the fact that there are only finite states, and by pigeon hole principle there must be at-least one state that gets repeated if the automaton runs on a long enough string. We also saw how to use this observation to produce new strings that should be accepted by the automaton. We abstracted this out as a lemma thus stating the pumping lemma.

In this lecture we provide an alternate view towards automata as a machine and present a beautiful theory which gives some fundamental understanding about the automata. The results are due to John Myhill and Anil Nerode, in 1958.

We view an automata as a classifier of strings in Σ^* based on which state it lands up if it is run on each string. More formally, let $M = (Q, \Sigma, \delta, s, F)$ be a finite state automaton accepting a language A. We define the following relation.

$$\forall x, y \in \Sigma^* : xRy \iff \hat{\delta}(s, x) = \hat{\delta}(s, y)$$

We study this relation more carefully and observe first that it is an equivalence relation.

- Reflexive: : $\forall x \in \Sigma^*$: xRx. That is, $\hat{\delta}(s, x) = \hat{\delta}(s, x)$.
- Symmetric : $\forall x, y \in \Sigma^*$: $xRy \Rightarrow yRx$. Let xRy. That is, $\hat{\delta}(s, x) = \hat{\delta}(s, y)$, hence $\hat{\delta}(s, y) = \hat{\delta}(s, x)$.
- Transitive : $\forall x, y, z \in \Sigma^* : xRy \wedge yRz \Rightarrow xRz$. Let xRy and yRz, $\hat{\delta}(s, x) = \hat{\delta}(s, y)$ and $\hat{\delta}(s, y) = \hat{\delta}(s, z)$ and hence $\hat{\delta}(s, x) = \hat{\delta}(s, z)$. Thus xRz.

Since it is a an equivalence relation, let us change notation to conform to standards. We will denote R by $\equiv_M$ (because it is defined based on M).

Is $\equiv_M$ merely an equivalence relation? Are there nicer properties that it satisfies. Note that we used only properties about "equality" in the above proofs?. Now we will use the fact that these are defined using $\hat{\delta}$.

- Right congruent: Suppose we run the automaton M on string x and y independently starting from the start state. The strings will be in the same equivalence class defined by $\equiv_M$ if we land up in the same state. What if we run the machine on xa and ya for an $a \in \Sigma$? Indeed, our intuition tells us that they should land up in the same state. To formally state this :

  $$\forall x, y \in \Sigma^* \, (x \equiv_M y \Rightarrow \forall a \in \Sigma : xa \equiv_M ya)$$

  We will formally prove this. Let $x, y \in \Sigma^*$ such that $x \equiv_M y$. That is, by definition of $\equiv_M$, we have $\hat{\delta}(s, x) = \hat{\delta}(s, y)$. Let $a \in \Sigma$. We want to prove that $\hat{\delta}(s, xa) = \hat{\delta}(s, ya)$. To do this, let us understand the LHS first.

  $$\begin{array}{rcl} \hat{\delta}(s, xa) & = & \delta(\hat{\delta}(s, x), a) \\ & = & \delta(\hat{\delta}(s, y), a) \\ & = & \hat{\delta}(s, ya) \\ \Rightarrow & & xa \equiv_M ya \end{array}$$

  By using the above property as a base case, we can extend right congruence from the alphabet $a \in \Sigma$ to arbitrary strings $z \in \Sigma^*$. That is we show the following.

  Claim 1. *$\forall x, y \in \Sigma^*$, if $x \equiv_M y$, then $\forall z \in \Sigma^* : xz \equiv_M yz$.*

  Proof. We proceed by induction on $|z|$. Bases case is $|z| = 0$. That is, $z = \epsilon$. In this case it follows trivially. Checking one more base case, the above stated right congruence property is the claim when $|z| = 1$.

  Induction Step : Let the claim be true for $|z| = k$. We will prove it for $|z| = k + 1$. Let $x, y \in \Sigma^*$ such that $x \equiv_M y$. Let $z \in \Sigma^{k+1}$. Our aim is to show that $xz \equiv_M yz$. Let us write $z = wa$ where $|w| = k$ and $a \in \Sigma$. The induction hypothesis gives us : $xw \equiv_M yw$. That is, $\hat{\delta}(s, xw) = \hat{\delta}(s, xw)$.

  We start with the LHS of what we aim to show: $\hat{\delta}(s, xz) = \hat{\delta}(s, xwa) = \delta(\hat{\delta}(s, xw), a) = \delta(\hat{\delta}(s, yw), a) = \hat{\delta}(s, ywa) = \hat{\delta}(s, yz)$. This completes the induction. □

- Respects membership in A: We build the intuition first. For $x, y \in \Sigma^*$, if we have $x \equiv_M y$, it indeed means that they make the automaton land up in the same state (say q) when run from the starting state s. Hence they must either be both inside A (if $q \in F$) or both outside A (if $q \notin F$). More formally,

  $$\forall x, y \in \Sigma^* \; : \; x \equiv_M y \Rightarrow (x \in A \iff y \in A)$$

  To complete we also give the formal proof: Let $x, y \in \Sigma^*$ such that $x \equiv_M y$. We have that $\hat{\delta}(s, x) = \hat{\delta}(s, y)$. Now,

  $$\begin{array}{rcl} x \in A & \iff & \hat{\delta}(s, x) \in F \\ & \iff & \hat{\delta}(s, y) \in F \\ & \iff & y \in A \end{array}$$

- Finite index: Let us understand how many equivalence classes can there be in Σ^* for the relation $\equiv_M$. For each state q, the strings which land up in that state are put in the same equivalence class. Hence the number of equivalence classes is exactly the number of states in the machine M and hence is finite.

Thus for every automaton M, we have an equivalence relation on Σ^* defined by $\equiv_M$ as above. Interestingly, these properties can be talked about, without reference to the automaton as such, and just with reference to the language. We make the following definition.

Definition 2. An equivalence relation $\equiv$ on Σ^* is said to be **Myhill-Nerode relation** with respect to A, if it satisfies the following properties.

- $\forall x, y \in \Sigma^* \, (x \equiv y \Rightarrow \forall a \in \Sigma : xa \equiv ya)$.
- $\forall x, y \in \Sigma^* \; : \; x \equiv y \Rightarrow (x \in A \iff y \in A)$.
- $\equiv$ is of finite index. The number of equivalence classes is finite.

We denote the fact that it is with respect to A by denoting $\equiv_A$.

Lemma 3. *If a language A is regular, then there is a Myhill-Nerode relation on Σ^* with respect to A.*

Proof. Let A be a regular language. Thus, there is an automaton M such that $A = L(M)$. Consider the relation $\equiv_M$. As we have already proved, $\equiv_M$ is a Myhill-Nerode relation with respect to $L(M)$ since it satisfyies all the above properties. □

This can be used to prove that certain languages are not regular. For example, to prove that a language $A \subseteq \Sigma^*$ is not regular, it suffices to to prove that there cannot be a Myhill-Nerode relation on Σ^* with respect to A. In other words, if an equivalence relation on Σ^* satisfies the first two conditions of the above definition, then it is not of finite index.

We will demonstrate this strategy for our favourite example for a non-regular language.

Theorem 4. *The language $A = \{a^n b^n : n \geq 0\}$ is not regular.*

Proof. We argue that there cannot be a Myhill-Nerode relation on Σ^* respecting the language A. Hence by the above lemma we will be able to conclude that A cannot be regular.

Our proof is by contradiction. Suppose $\equiv$ is a Myhill-Nerode relation on Σ^* with respect to A. It satisfies the above properties : the right congruence property (hence it also satisfies claim 1), respects the membership of A, and is of finite index.

Consider k and m, $k \neq m$. Consider the strings $x = a^k$ and $y = a^m$. Are they in the same equivalence class by $\equiv$? We argue that they cannot be. Let $z = b^k$. Thus $xz = a^k b^k$, and $yz = a^m b^k$. Thus $xz \in A$ and $yz \notin A$. Since the equivalence relation respects membership in A, it has to be that $xz \not\equiv yz$. But then, if we apply claim1 for this $z = b^k$, we conclude that $a^k \not\equiv a^m$. Since there are infinitely many choices of k and m such that $k \neq m$, we conclude that there has to be infinitely many equivalence classes.

□

Exercise : Try a similar proof for the language $A = \{a^p : p \text{ is a prime number } \}$.