CMPSCI 250: Introduction to Computation

Lecture #32: The Myhill-Nerode Theorem
David Mix Barrington
11 April 2014

The Myhill-Nerode Theorem

- Review: L-Distinguishable Strings
- The Language Prime has no DFA
- The Relation of L-Equivalence
- More Than k Classes Means More Than k States
- Constructing a DFA From the Relation
- Completing the Proof
- The Minimal DFA and Minimizing DFA's

Review: L-Distinguishable Strings

- Let $L \subseteq \Sigma^*$ be any language. Two strings u and v are **L-distinguishable** (or **L-inequivalent**) if there exists a string w such that $uw \in L \oplus vw \in L$.
- They are **L-equivalent** if for every string w, $uw \in L \leftrightarrow vw \in L$ (we write this as $u \equiv_L v$).
- We proved last time that if a DFA takes two L-distinguishable strings to the same state, it cannot have L as its language.

Clicker Question #1

- Let $\Sigma = \{a, b\}$ and X be the language $(\Sigma^3)^*$, which is the set of all strings whose length is divisible by 3. Which one of these pairs of strings is X-distinguishable?
- (a) abab and abaaabb
- (b) aabbbbaba and λ
- (c) abba and abaa
- (d) b and bbabb

Answer #1

- Let $\Sigma = \{a, b\}$ and X be the language $(\Sigma^3)^*$, which is the set of all strings whose length is divisible by 3. Which one of these pairs of strings is X-distinguishable?
- (a) abab and abaaabb
- (b) aabbbbaba and λ
- (c) abba and abaa
- *(d) b and bbabb (append b, for example)*

L-Distinguishable Strings

- We use this fact to prove a lower bound on the number of states in a DFA for L. Suppose we can find a set S of k strings that are *pairwise* L-distinguishable. Then it is impossible for a DFA with *fewer than k* states to have L as its language.
- If S is an *infinite* set of pairwise L-distinguishable strings, no correct DFA for L can exist at all.

The Paren Language

- For example, the language Paren $\subseteq \{L, R\}^*$ has such a set, $\{L^i : i \geq 0\}$, because if $i \neq j$ then L^iR^i is in Paren but L^jR^i is not.
- So any two distinct strings in the set are L-distinguishable.
- No DFA for Paren exists, and thus Paren is not a regular language.

Prime Has No DFA

- Let Prime be the language $\{a^n: n \text{ is a prime number}\}$. It doesn't seem likely that any DFA could decide Prime, but this is a little tricky to prove.
- Let i and j be two naturals with $i > j$. We'd like to show that a^i and a^j are Prime-distinguishable, by finding a string a^k such that $a^i a^k \in$ Prime and $a^j a^k \notin$ Prime (or vice versa).
- We need a natural k such that $i + k$ is prime and $j + k$ not, or vice versa.

Prime Has No DFA

- Pick a prime p bigger than both i and j (since there are infinitely many primes).
- Does k = p - j work? It depends on whether i + (p - j) is prime -- if it isn't we win because j + (p - j) is prime. If it is prime, look at k = p + i - 2j. Now j + k is the prime p + (i - j), so if i + k = p + 2(i - j) is not prime we win.
- We find a value of k that works unless all the numbers p, p + (i - j), p + 2(i - j),..., p + r(i - j),... are prime. But p + p(i - j) is not prime as it is divisible by p.

The Relation of L-Equivalence

- The relation of L-equivalence is aptly named because we can easily prove that it is an equivalence relation.
- Clearly $\forall w$: $uw \in L \leftrightarrow uw \in L$, so it is reflexive.
- If we have that $\forall w$: $uw \in L \leftrightarrow vw \in L$, we may conclude that $\forall w$: $vw \in L \leftrightarrow uw \in L$, and thus it is symmetric.
- Transitivity is equally simple to prove.

Clicker Question #2

- Again let $\Sigma = \{a, b\}$ and let $X = (\Sigma^3)^*$. Which one of these sets of strings is pairwise X-inequivalent, and thus contains one element of each X-equivalence class?
- (a) $\{\lambda, a, b\}$
- (b) $\{\lambda, aaa, aab, abb, bbb\}$
- (c) $\{\lambda, b, bb, bbb\}$
- (d) $\{\lambda, aa, abbbabb\}$

Answer #2

- Again let $\Sigma = \{a, b\}$ and let $X = (\Sigma^3)^*$. Which one of these sets of strings is pairwise X-inequivalent, and thus contains one element of each X-equivalence class?
- (a) $\{\lambda, a, b\}$
- (b) $\{\lambda, aaa, aab, abb, bbb\}$
- (c) $\{\lambda, b, bb, bbb\}$
- *(d) $\{\lambda, aa, abbbabb\}$*

The Myhill-Nerode Theorem

- We know that any equivalence relation partitions its base set into equivalence classes.
- The **Myhill-Nerode Theorem** says that for any language L, there exists a DFA for L with k or fewer states if and only if the L-equivalence relation's partition has k or fewer classes.

The Myhill-Nerode Theorem

- That is, if the number of classes is a natural k then there is a minimal DFA with k states.
- If the number of classes is infinite then there is no DFA at all.
- It's easiest to think of the theorem in the form: "k or fewer states $\leftrightarrow$ k or fewer classes".

(≥ k Classes) → (≥ k States)

- We've essentially already proved half of this theorem. We can take "k or fewer states → k or fewer classes" and take its contrapositive, to get "more than k classes → more than k states".
- Let L be an arbitrary language and assume that the L-equivalence relation has more than k (non-empty) equivalence classes. Let $x_1,\ldots,x_{k+1}$ be one string from each of the first k + 1 classes.
- Since any two distinct strings in this set are in different classes, by definition they are not L-equivalent, and thus they are L-distinguishable.

(≥ k Classes) → (≥ k States)

- By our result from last lecture, since there exists a set of k + 1 pairwise L-distinguishable strings, no DFA with k or fewer states can have L as its language.
- This proves the first half of the Myhill-Nerode Theorem.
- The second half will be a bit more complicated.

Making a DFA From the Relation

- Now to prove the other half, "k or fewer classes → k or fewer states".
- In fact we will prove that if there are exactly k classes, we can build a DFA with exactly k states.
- This DFA will necessarily be the smallest possible for the language, because a smaller one would contradict the first half of the theorem, which we have just proved.

Making a DFA From the Relation

- Let L be an arbitrary language and assume that the classes of the relation are $C_1, \ldots, C_k$. We will build a DFA with states $q_1, \ldots, q_k$, each state corresponding to one of the classes.
- The initial state will be the state for the class containing λ. The final states will be any states that contain strings that are in L. The transition function is defined as follows. To compute $\delta(q_i, a)$, where $a \in \Sigma$, let w be any string in the class C_i and define $\delta(q_i, a)$ to be the state for the class containing the string wa.

Making a DFA From the Relation

- It's not obvious that this δ function is well-defined, since its definition contains an arbitrary choice. We must show that any choice yields the same result.
- Let u and v be two strings in the class C_i. We need to show that ua and va are in the same class as each other.
- That is, for any u, v, and a, we must show that $(u \equiv_L v) \rightarrow (ua \equiv_L va)$.

The δ Function is Well-Defined

- Assume that $\forall w$: $uw \in L \leftrightarrow vw \in L$.
- Let z be an arbitrary string.
- Then $uaz \in L \leftrightarrow vaz \in L$, because we can specialize the statement we have to az.
- We have proved that $\forall z$: $uaz \in L \leftrightarrow vaz \in L$, which by definition means that $ua \equiv_L va$.

Completing the Proof

- Now we prove that for this new DFA and for any string w, $\delta^*(i, w) = q_j \leftrightarrow w \in C_j$. (Here "i" is the initial state of the DFA.)
- We prove this by induction on w. Clearly $\delta^*(i, \lambda) = i$, which matches the class of λ.
- Assume as IH that $\delta^*(i, w) = x$ matches the class of w. Then for any a, $\delta^*(i, wa)$ is defined as $\delta(x, a)$, which matches the class of wa by the definition, which is what we want.

Completing the Proof

- If two strings are in the same class, either both are in L or both are not in L.
- So L is the union of the classes corresponding to our final states.
- Since the DFA takes a string to the state for its class, $\delta^*(i, w) \in F \leftrightarrow w \in L$.
- Thus this DFA decides the language L.

Clicker Question #3

- Again let $\Sigma = \{a, b\}$ and let $X = (\Sigma^3)^*$. We saw earlier that there are three X-equivalence classes, so the MN theorem gives us a DFA for X with three states. Which statement about this DFA is false?
- (a) The class of λ is final and the other two are not.
- (b) The a-arrow and b-arrow from a given state s always both go to the same state t.
- (c) The b-arrow from the class of a goes to itself.
- (d) The initial state is for the class of λ.

Answer #3

- Again let $\Sigma = \{a, b\}$ and let $X = (\Sigma^3)^*$. We saw earlier that there are three X-equivalence classes, so the MN theorem gives us a DFA for X with three states. Which statement about this DFA is false?
- (a) The class of λ is final and the other two are not.
- (b) The a-arrow and b-arrow from a given state s always both go to the same state t.
- *(c) The b-arrow from the class of a goes to itself.*
- (d) The initial state is for the class of λ.

The Minimal DFA

- Let X be a regular language and let M be *any* DFA such that L(M) = X.
- We will show that the minimal DFA, constructed from the classes of the L-equivalence relation, is **contained within** M.
- We begin by eliminating any unreachable states of M, which does not change M's language.

The Minimal DFA

- Remember that a correct DFA cannot take two L-distinguishable strings to the same state.
- So for any state p of M, the strings w such that $\delta(i, w) = p$ are all L-equivalent to each other.
- Each state of M is thus associated with one of the classes of the L-equivalence relation.

Minimizing a DFA

- The states of M are thus partitioned into classes themselves.
- If we combine each class into a single state, we get the minimal DFA.
- In discussion on Monday we will see, and then practice, a specific algorithm that will find these classes. It thus will construct the minimal DFA equivalent to any given DFA.