Chapter 5
Applications of the Myhill–Nerode Theorem

This chapter is devoted to justifying our praise for the Myhill–Nerode theorem, by developing a few of its applications. We strive to display both the usefulness of the theorem and its versatility.

The usefulness of the Myhill–Nerode theorem. In Section 5.1, we show how to use the theorem to prove that certain languages are not regular. Indeed, we argue, both in that section and in Section 6.1.2 that the Myhill–Nerode theorem is always the preferred tool for this purpose.

In Section 5.2, we use the theorem to develop an algorithm for minimizing the number of states in an FA. The input to the algorithm we develop is a FA $M$; the output is a FA $M'$ that is equivalent to $M$, in the sense that $L(M') = L(M)$, and that has the smallest number of states of any FA that is equivalent to $M$. This algorithm is usually part of every college curriculum in computer or electrical engineering, but the theoretical underpinnings of the algorithm—namely, the theorem—are, regrettably, seldom taught.

The far-reaching implications of the Myhill–Nerode theorem. In Section 5.3, we use the theorem to analyze aspects of a probabilistic FA-like model from [77], which, despite its 1963 vintage, shares many of the characteristics of models that are used in modern studies of machine (computer) learning. The main result of the section shows that a certain class of these models accept only regular languages—so that their computational power is not enhanced by allowing probabilistic state transitions. In Section 5.4, we use the theorem to derive a lower bound, first observed in [47], on the amount of memory that is needed to decide membership in any nonregular language. The bound assumes the following form. Let $L$ be a nonregular language. Then for infinitely many integers $n$, any FA that correctly decides membership/nonmembership in $L$ of all words of length $\leq n$ must have no fewer than $f(n)$ states. The bound $f(n)$ is derived via an analogue of the equivalence relation $\equiv_L$ of the Myhill–Nerode theorem. Finally, in Section 5.5, we use the theorem to derive a lower bound, first observed in [38], on the time needed by an online TM that has $t$ tapes, each of dimensionality $d$, to solve a simplified database problem. By considering algorithms for solving this problem on other models, one obtains significant
information about the computational implications of the “online” regimen for computing and about the relative powers of a variety of families of data structures.

5.1 Proving that Languages Are Not Regular

Finite automata are very limited in their computing power due to the finiteness of their memories, i.e., of their sets of states. Indeed, as one might infer from the Myhill–Nerode theorem, the standard way to expose the limitations of FAs—by proving that a language $L$ is not regular—is to establish somehow that the structure of $L$ requires distinguishing among infinitely many distinct situations.

The finite-index lemma and fooling sets. Given the conceptual parsimony and power of Theorem 4.1, it is not surprising that the theorem affords one a simple, yet powerful, tool for proving that a language is not regular. This tool is encapsulated in the following lemma, which is an immediate corollary of the equivalence of statements (1) and (3) in the theorem, and which can be viewed as a strengthening of the continuation lemma for OAs (Lemma 3.1). For reasons that we hope will become suggestive imminently, we refer to the upcoming lemma as “the finite-index lemma.”

We maintain that the ensuing development should be viewed as the primary tool for proving that a language is not regular.

Lemma 5.1. (The finite-index lemma) Let $L \subseteq \Sigma^*$ be an infinite regular language. Every sufficiently large set of words over $\Sigma$ contains at least two distinct words, $x$ and $y \neq x$, such that $x \equiv_L y$.

Proof. Let us say that the infinite regular language $L$ is accepted by the FA $M = (Q, \Sigma, \delta, q_0, F)$. Because the set $Q$ is finite, any infinite set of words from $\Sigma^*$ must—by the pigeonhole principle$^1$—always contain two distinct words, $x$ and $y \neq x$, that are indistinguishable to $M$, in the sense that $x \equiv_M y$. Clearly, then, $x \equiv_L y$; cf. Lemma 3.2. (The validating argument proceeds as follows. Let us enumerate $\Sigma^*$ in some way—the specific way is not relevant to the argument—and note, for each word $w \in \Sigma^*$ in our enumeration, the state $\delta(q_0, w)$ to which $w$ leads $M$. Because $Q$ is finite, we must eventually find distinct words, $x$ and $y$, such that $\delta(q_0, x) = \delta(q_0, y)$. By definition, $x \equiv_M y$.)

A direct calculation based on the way we extended the state-transition function $\delta$ to the domain $Q \times \Sigma^*$ now verifies that

$$\forall z \in \Sigma^* \left[ \delta(q_0, xz) = \delta(\delta(q_0, x), z) = \delta(\delta(q_0, y), z) = \delta(q_0, yz) \right].$$

(We proceed from expression #2 in this chain to expression #3 via the universal algebraic operation of substituting equals for equals.) This system of equalities means that $x \equiv_L y$. □

$^1$ The “pigeonhole principle” asserts: If you put $n + 1$ balls into $n$ bins, some bin must receive more than one ball. It is sometime called “Dirichlet’s box principle,” after Johann P.G. Lejeune Dirichlet.