NON-REGULAR LANGUAGES, THE MYHILL-NERODE THEOREM, AND LINEAR ALGEBRA TESTS

JOEL FRIEDMAN

Contents

Disclaimer: The material may sketchy and/or contain errors, which I will elaborate upon and/or correct in class. For those not in CPSC 421/501: use this material at your own risk...

There are a number of ways to test whether or not a language, L, is regular, and—if so—what is the smallest number of states that a DFA recognizing L can have. Most techniques we know can be organized as follows:

(1) results on languages over an alphabet of size one, new to this article as of Fall 2021[1];
(2) "derived results," where proven results on languages give similar results on related languages;
(3) the Myhill-Nerode Theorem;
(4) consequences of linear algebra, applied to the adjacency matrix of a DFA; and
(5) the Pumping Lemma.

Methods (1,2,4) are the easiest to apply; in recent years I have not covered (4), because (4) requires linear algebra that is not a prerequisite for CPSC 421/501.[2]

Date: Wednesday 11th October, 2023, at 09:30(get rid of time in final version).
Research supported in part by an NSERC grant.

[1] Based on class discussion of September 28, 2021, and, in particular a question of Markus de Medeiros.

[2] There is a strong argument that a first-term linear algebra course is more relevant to computer science than, say, a second-term calculus course; we thank Vee Kay for discussions on this point.

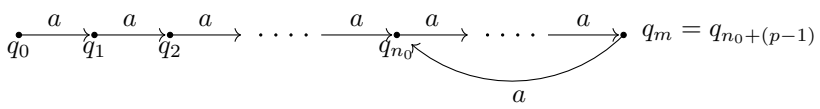


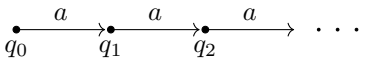
FIGURE 1. A DFA over $\Sigma = \{a\}$ with period p.

Much of this article is devoted to method (3), the Myhill-Nerorde theorem, which—in principle—always tells you whether or not a language is regular, and, if so, the minimum number of states needed in a DFA that recognizes it.

Method (5), the Pumping Lemma, is by far the most awkward technique to use; its only advantage is that there is an analogous Pumping Lemma for context-free languages. In CPSC 421 this year we are skipping over the chapter on context-free languages, so we have little motivation to cover the Pumping Lemma. Moreover, any result that can be proven with the Pumping Lemma can be proven (usually more simply and directly) using the Myhill-Nerode theorem.

1. Languages Over an Alphabet Consisting of a Single Letter

In class we explain that if $M = (Q, \Sigma, \delta, q_0, F)$ is a DFA over a language where Σ consists of a single letter, then the DFA begins in state q_0 and begins with some number of new states:



Since the DFA has only finitely many states, at some point there is a state q_m that points to a previous state, namely q_{n_0} with $0 \le n_0 \le m$; we depict this in Figure 1. Notice that all strings in $\{a\}^*$ must land in one of states $q_0, \ldots, q_m$, and hence—for the sake of recognizability—we may discard any additional states in the DFA. The equivalent Figure 2 may be more suggestive of the geometry: a *path* of length n_0 (i.e., n_0 edges) followed by a *cycle* of length p (i.e. p edges).[3]

Definition 1.1. Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA where $\Sigma = \{a\}$, and is of the form depicted in Figure 2 (i.e., we have discarded any irrelevant states from M beyond $q_0, \ldots, q_m$). We refer to p as the *cycle length* of M, and to n_0 as the *cycle start point* of M.

Let M be a DFA over $\{a\}$, with p, n_0 as depicted in Figure 2 (hence discard or ignore any irrelevant states of M). It is clear that for $n < n_0 + p$, the string a^n is taken to the state q_n in this DFA. Since M has a cycle of length p traversing the vertices

$$q_{n_0}, q_{n_0+1}, \ldots, q_m = q_{n_0+(p-1)}, q_{n_0},$$

it is clear that the strings

$$a^{n_0}, a^{n_0+p}, a^{n_0+2p}, a^{n_0+3p}, \ldots$$

[3] The terms *path* and *cycle* are notions in *graph theory*, and make sense both for *directed graphs* and *graphs* (i.e., *undirected graphs*); for the definition of graphs and digraphs (i.e., directed graphs), see [Sip], Chapter 0 (page 10 in the 3rd edition of [Sip]). We will use the terms *path* and *cycle* in their intuitive meaning (as often done in graph theory), and leave their formal definition to the reader (who may look up these definitions elsewhere).

are all taken to state q_{n_0} in this DFA. Similarly, for any $i = 0, 1, \ldots, p-1$,

$$a^{n_0+i}, a^{n_0+i+p}, a^{n_0+i+2p}, a^{n_0+i+3p}, \ldots$$

are all taken to state q_{n_0+i} in this DFA.

Hence we have proven the following.

Proposition 1.2. *Let $M = (Q, \Sigma, \delta, q_0, F)$ be a DFA with $\{a\}$. L be the language recognized by M. Then for some integers $n_0 \geq 0$ and $p \geq 1$, M has $n_0 + p$ states, and for all $n \geq n_0$ we have*

$$a^n \in L \quad \iff \quad a^{n+p} \in L.$$

In other words, for any $i = 0, 1, \ldots, p-1$ we have that

$$a^{n_0+i}, a^{n_0+i+p}, a^{n_0+i+2p}, a^{n_0+i+3p}, \ldots$$

either (1) all lie in L or (2) all lie outside L. In other words, for $n \geq n_0$, whether or not a^n lies in L or outside L depends only on the value of $n \bmod p$ (which is between 0 and $p-1$).

Of course, the analogous proposition is true for any Σ consisting of a single symbol.

We therefore deduce the following theorem.

Definition 1.3. Let $p \in \mathbb{N}$. We say that a language, L, over $\Sigma = \{a\}$ is *eventually p-periodic* if for some integer $n_0 \geq 0$ we have

$$\forall n \geq n_0, \quad a^n \in L \iff a^{n+p} \in L. \tag{1}$$

If so, we also say that L is *eventually periodic*. If so, the *period of L* is the smallest p for which L is p-periodic.

Theorem 1.4. *A language, $L \subset \Sigma^*$, over $\Sigma = \{a\}$ is regular iff it is* eventually periodic.

It is not hard to see that if a language L over $\Sigma = \{a\}$ is eventually periodic, then DFA accepting L with the smallest number of states is the DFA depicted in Figure 1 where p is the period of L and n_0 is the smallest integer such that (1) holds (see Exercise 6.1.2).

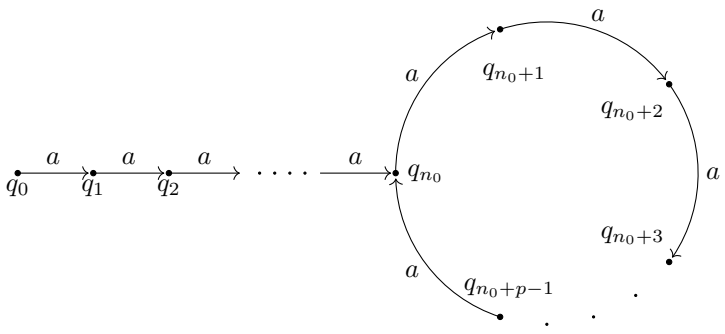


FIGURE 2. Figure 1 Redrawn to Emphasize the Cycle

Example 1.5. Let

$$L = \{a^{(n^3)} \mid n \in \mathbb{N}\} = \{a, a^8, a^{27}, a^{64}, \ldots\}$$

be the language of strings of a's whose length is a perfect cube. Then since the interval between any two perfect cubes is

$$(n+1)^3 - n^3 = 3n^2 + 3n + 1,$$

which is arbitrarily large, L cannot be p periodic (since for any fixed p and large n, $a^{n^3+1}, a^{n^3+2}, \ldots, a^{n^3+p}$ are not in L, which would imply that L is finite).

One can similarly show that if $n_1 < n_2 < \cdots$ is an infinite (increasing) sequence of non-negative integers, then if

$$L = \{a^{n_i} \mid i \in \mathbb{N}\} = \{a^{n_1}, a^{n_2}, \ldots\}$$

is regular, we must have $n_{i+1} - n_i$ must be bounded above over all i (see Exercise 6.1.8).

2. Derived Results: Part 1

Here is a standard way of producing more lower bounds (or examples of non-regular languages) from others.

Proposition 2.1. *Let $L_1, L_2 \subset \Sigma^*$ be lanuages over Σ. If L_1 is regular, and $L_1 \cap L_2$ is non-regular, then L_2 is non-regular.*

This follows immediately from the fact that the intersection of any two regular languages is, again, regular.

Example 2.2. Let $L \subset \Sigma^*$ be the language over $\Sigma = \{a, b\}$ of strings/words whose length is a perfect cube. Then L is non-regular, since $L \cap \{a\}^*$ is, according to Example 1.5, non-regular.

3. The Myhill-Nerode Theorem: Part 1

Definition 3.1. If L is a language over an alphabet Σ, and $s \in \Sigma^*$, we define the *accepting futures of s in L* to be

$$\text{AcceptingFuture}_L(s) \stackrel{\text{def}}{=} \{s' \in \Sigma^* \mid ss' \in L\}.$$

We will also use the abbreviation AccFut.

Example 3.2. Let $\Sigma = \{0, 1, \ldots, 9\}$, and

$$L = \text{DIV-BY-2} = \{0, 2, 4, 6, 8, 10, 12, \ldots\}.$$

In class we gave a 5 state DFA that recognizes this language. We have

$$\begin{aligned}
\text{AccFut}_L(\epsilon) &= L = \{0, 2, 4, 6, 8, 10, \ldots\} \\
\text{AccFut}_L(0) &= \{\epsilon\} \\
\text{AccFut}_L(00) &= \emptyset \\
\text{AccFut}_L(1) &= \Sigma^*(0, 2, 4, 6, 8) \\
\text{AccFut}_L(2) &= \{\epsilon\} \cup \Sigma^*(0, 2, 4, 6, 8)
\end{aligned}$$

Note that we have

$$\text{AccFut}_L(2) = \text{AccFut}_L(4) = \text{AccFut}_L(2238) = \cdots,$$

so many strings have the same accepting future with respect to L.

In class we similarly gave 6 different strings with different accepting futures for the language DIV-BY-3. We also explained why if DIV-BY-3 has 6 different accepting futures, then any DFA recognizing DIV-BY-3 must have at least 6 different states. More generally we have the following observation.

Proposition 3.3. *If a language, L, over an alphabet, Σ, has at least n distinct accepting futures (i.e., n distinct values of $\mathrm{AccFut}_L(s)$ with $s \in \Sigma^*$), then any DFA recognizing L has at least n states.*

Proof. If $s, s' \in \Sigma^*$ are taken to the same state in a DFA, then for any $t \in \Sigma^*$, st lands in an accepting state of the DFA iff $s't$ does. Hence if

$$\mathrm{AccFut}_L(s_1), \ldots, \mathrm{AccFut}_L(s_n)$$

are distinct, then a DFA recognizing L must take $s_1, \ldots, s_n$ to distinct states. □

Example 3.4. Let $\Sigma = \{0, 1\}$ and

$$L = \{0^n 1^n \mid n \in \mathbb{N}\} = \{01, 0011, 000111, 0^4 1^4, \ldots\}$$

we have

$$\begin{aligned} \mathrm{AccFut}_L(0) &= \{1, 011, 00111, \ldots\} \\ \mathrm{AccFut}_L(00) &= \{11, 0111, 001111, \ldots\} \\ \mathrm{AccFut}_L(000) &= \{111, 01111, \ldots\} \end{aligned}$$

and, more generally, $\mathrm{AccFut}_L(0^k)$ has a unique shortest string, namely 1^k. Hence $\mathrm{AccFut}_L(0^k)$ for $k \in \mathbb{N}$ are all distinct, and so L is not regular.

4. The Myhill-Nerode Theorem: Part 2

The second part of the Myhill-Nerode is a converse to the proposition in the last section.

Theorem 4.1. *Let L be a language over an alphabet Σ, and assume that there is a finite number, n, of distinct values of*

$$\mathrm{AccFut}_L(s)$$

as s varies over Σ^*. Then there exists a DFA with n states that recognizes L.

Actually, much more is true in the above theorem: one can actually build the DFA, and to do so one does not need to describe all of $\mathrm{AccFut}_L(s)$ for strings, s—rather, one needs only to be able to tell for $s, s' \in \Sigma^*$ whether or not $\mathrm{AccFut}_L(s)$ equals $\mathrm{AccFut}_L(s')$.

To prove the theorem we consider the languages:

(1) $\mathrm{AccFut}_L(\epsilon)$;
(2) $\mathrm{AccFut}_L(a)$, $\mathrm{AccFut}_L(b)$;
(3) $\mathrm{AccFut}_L(aa)$, $\mathrm{AccFut}_L(ab)$, $\mathrm{AccFut}_L(ba)$, $\mathrm{AccFut}_L(bb)$;
(4) etc.

Each time we see a new language, i.e., a new value of $\mathrm{AccFut}_L(s)$, we introduce a new state; the transition rule $\delta\colon Q \times \Sigma \to Q$ is given by

$$\delta(\mathrm{AccFut}_L(s), \sigma) = \mathrm{AccFut}_L(s\sigma)$$

(where we identify a value of $\mathrm{AccFut}_L(s)$ with its corresponding state). It is much easier to understand the theorem and its proof from an example.

Example 4.2. Let $\Sigma = \{a, b\}$ and

$$L = \{s \in \Sigma^* \mid s \text{ contains } ab \text{ as a substring}\}.$$

We begin by computing

$$\mathrm{AccFut}_L(\epsilon) = L,$$

which we associate with a state q_0 and to the string ϵ; we then comptue

$$\mathrm{AccFut}_L(a) = b\Sigma^* \cup L, \quad \mathrm{AccFut}_L(b) = L, \tag{2}$$

which gives us a new state, q_1 associated to $b\Sigma^* \cup L$ and associate to the input string a; we do not introduce a new state for b, since we have already seen the language $\mathrm{AccFut}_L(b) = L$ associated to q_0 and ϵ.

At this point:

(1) We have determined two states, q_0, q_1, of our DFA;
(2) q_0 is the initial state of the DFA, since the initial state is the state you reach on input ϵ;
(3) from q_0, associated to ϵ, based on (2) we have the transition rules

$$\delta(q_0, a) = q_1, \quad \delta(q_0, b) = q_0. \tag{3}$$

As a next step we want to determine $\delta(q_1, \sigma)$ for $\sigma = a, b$. We compute that

$$\mathrm{AccFut}_L(aa) = b\Sigma^* \cup L, \quad \mathrm{AccFut}_L(ab) = \Sigma^*;$$

since we have already encountered $b\Sigma^* \cup L$ but not Σ^*, we introduce a new state q_2 associated to ab and to Σ^*, and declare

$$\delta(q_1, a) = q_1, \quad \delta(q_1, b) = q_2. \tag{4}$$

Next we want to determine $\delta(q_2, \sigma)$ for $\sigma = a, b$. We compute that

$$\mathrm{AccFut}_L(aba) = \Sigma^*, \quad \mathrm{AccFut}_L(abb) = \Sigma^*;$$

since we have already seen Σ^*, which is associated to q_2, we declare

$$\delta(q_2, a) = q_2, \quad \delta(q_2, b) = q_2. \tag{5}$$

At this point we have determined $\delta(q, \sigma)$ for all $\sigma = a, b$ and all states, i.e., q_0, q_1, q_2, without introducing new states. Hence $Q = \{q_0, q_1, q_2\}$, and δ is given by (3)–(5) above.

Finally, since

$$\epsilon \notin L, \quad \epsilon \notin b\Sigma^* \cup L, \quad \epsilon \in \Sigma^*,$$

the state q_2 is an accepting (or final) state, and q_0, q_1 are not. (The general principle here is that $\epsilon \in \mathrm{AccFut}_L(s)$ iff $s \in L$.) This determines the DFA.

Abstractly, the reason why the above construction works is that if

$$\mathrm{AccFut}_L(s) = \mathrm{AccFut}_L(s')$$

for some s, s', then for any $\sigma \in \Sigma$ we have

$$\mathrm{AccFut}_L(s\sigma) = \mathrm{AccFut}_L(s'\sigma).$$

In the above example we have

$$\mathrm{AccFut}_L(a) = \mathrm{AccFut}_L(aa),$$

and this implies that for $\sigma = a, b$ we have

$$\mathrm{AccFut}_L(a\sigma) = \mathrm{AccFut}_L(aa\sigma);$$

hence the transition from the state of aa to that of $aa\sigma$ is the same as of that from a to $a\sigma$.

Notice that in the above construction we don't need to determine *all of* $\mathrm{AccFut}_L(s)$ for strings, $s \in \Sigma^*$; it suffices to know for certain $s, s' \in \Sigma^*$ whether or not $\mathrm{AccFut}_L(s)$ and $\mathrm{AccFut}_L(s')$ are equal.

5. Derived Results: Part 2

Once we prove that certain langauges are not regular, we can infer that other languages are not regular. Here is one such examples of a "derived result."

Above we have proven that $L = \{0^n 1^n \mid n \in \mathbb{N}\}$ is not regular; this is also done in the textbook by Sipser (and similar textbooks) using the Pumping Lemma. Consider the language

$$L' = \big\{s \in \{0,1\}^* \mid s \text{ has the same number of 0's and 1's}\big\}.$$

Then L' is not regular, for if L' were regular then

$$L' \cap 0^*1^*$$

would also be regular, which is impossible since $L = L' \cap 0^*1^*$.

Note that the Myhill-Nerode theorem gives a more direct proof that L' is not regular: for any $k \in \mathbb{N}$, $\mathrm{FutAcc}_{L'}(0^k)$ has a unique shortest length string, which is 1^k; hence the languages $\mathrm{FutAcc}_{L'}(0^k)$ are distinct for $k \in \mathbb{N}$, and so the Myhill-Nerode theorem implies that L' is not regular.

In the the appendix we will also explain how to use linear algebra tests to show that L' is not regular.

6. EXERCISES

6.1. Exercises on Languages over $\Sigma = \{a\}$.

For all these exercises, recall that $\mathbb{Z}_{\geq 0}$ refers to the set $\{0, 1, 2, \ldots\}$, i.e., the set of non-negative integers.

Unless otherwise instructed, DO NOT USE THE MYHILL-NERODE THEOREM for the exercises in this subsection. You can assume the results of Section 1, but **do not assume anything further, such as consequences of this section that we discussed in class; do not merely quote results stated in class.**

Exercise 6.1.1. Let L be a language over the alphabet $\Sigma = \{a\}$. Let $m \in \mathbb{Z}_{\geq 0}$.

6.1.1(a) Explain why if L is finite and $a^m \in L$, then any DFA accepting L must have at least $m + 2$ states.

6.1.1(b) Explain why if L contains $a^{m+1}, a^{m+2}, a^{m+3} \ldots$, but does not contain a^m, then any DFA accepting L must have at least $m + 2$ states.

Exercise 6.1.2. Let $L \subset \Sigma^*$ be a regular language with $\Sigma = \{a\}$.

6.1.2(a) Show that for any positive naturals $m < m'$, if L is eventually m-periodic and eventually m'-periodic, then L is eventually $(m' - m)$-periodic.

6.1.2(b) Let p be the period of L, i.e., the smallest positive integer such that L is eventually p-periodic. Show that if L is p'-periodic for some $p' \in \mathbb{N}$, then p' must be a multiple p.

6.1.2(c) Let p be the period of L. Show that the cycle length of any DFA that recognizes L has period divisible by p.

6.1.2(d) Let p be the period of L. Show that if M is a DFA that recognizes L and whose cycle length is larger than p, then there exists a DFA with fewer states than M that also recognizes L. [Hint: can you replace the cycle in M by a cycle of smaller length?]

6.1.2(e) Let p be the period of L. Explain why if n_0 is the smallest integer such that (1) holds, then the DFA with the fewest states that recognizing L has $n_0 + p$ states.

Exercise 6.1.3. Let $L = \{a^{3n} \mid n \in \mathbb{Z}_{\geq 0}\}$. What is the minimum number of states in a DFA needed to recognize L? **Explain this as briefly as possible.** Give such a DFA. You may use the results in Exercise 6.1.2.

Exercise 6.1.4. Let $L = \{a^{3n} \mid n \in \mathbb{N}\}$. What is the minimum number of states in a DFA needed to recognize L? **Explain this as briefly as possible.** Give such a DFA. You may use the results in Exercise 6.1.2.

Exercise 6.1.5. Let L be a regular language over the alphabet $\Sigma = \{a\}$ that contains all strings of sufficiently large even length. What are the possible values of the period of L? You may use the results in Exercise 6.1.2.

Exercise 6.1.6. Let

$$L = \{a^p \mid p \text{ is a prime number}\} = \{a^2, a^3, a^5, a^7, \ldots\}.$$

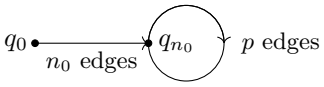
What does it mean to conjecture that $(a^2 L) \cap L$ is infinite? Identify this as a well-known conjecture in number theory.

Exercise 6.1.7. Let $n_1 < n_2 < n_3 < \cdots$ be an infinite sequence of non-negative integers, and let

$$L = \{a_{n_1}, a_{n_2}, \ldots\}$$

Show that if $n_{i+1} - n_i$ becomes arbitrarily large for $i \in \mathbb{N}$[4], then L is non-regular.

Exercise 6.1.8. Let $L \subset \{a\}^*$ be a regular language over the alphabet $\Sigma = \{a\}$, and let p be the period of L, and let n_0 be the smallest integer such that for all $n \geq n_0$, we have $a^n \in L$ iff $a^{n+p} \in L$. Hence the DFA, M, with the fewest states that recognizes L has $n_0 + p$ states. Recall that M looks like Figure 2, i.e., in brief:



Say that $a^k \in L$ for some k; the *forward gap of L at k* refers to the smallest $t \in \mathbb{N}$ such that $a^{k+t} \in L$; the *backwards gap of L at k* referes to the smallest $t' \in \mathbb{N}$ such that $a^{k-t'} \in L$. Assume that such t, t' exist and that $t \neq t'$ (i.e., the foward gap at k is does not equal the backward gap).

6.1.8(a) Show that if $k - t' \geq n_0$, then $p \geq t + t'$, and therefore the number of states in M is at least $t + t'$.

[4] This mean that for any $C \in \mathbb{Z}$ there is an $i \in \mathbb{Z}$ such that $n_{i+1} - n_i \geq C$.

6.1.8(b) Show that if $k - t' + 1 \leq n_0 \leq k$, then $p \geq t + 1$; conclude that $n_0 + p \geq k + t - t' + 2$, and hence M has at least $k + t - t' + 2$ states.

6.1.8(c) Show that if $k + t \leq n_0$, then M has at least $k + t$ states.

6.1.8(d) Hence conclude that M has at least

$$\min(t + t', k + t - t' + 2, k + t)$$

states.

Exercise 6.1.9. Let $L \subset \{a\}^*$ be an infinite, regular language over the alphabet $\Sigma = \{a\}$, such that $a^{100}, a^{110} \in L$, but $a^{111}, a^{112}, \ldots, a^{120} \notin L$.

6.1.9(a) Use Exercise 6.1.8 to show that any DFA recognizing L must have at least 21 states.

6.1.9(b) Give a language L that satisfies the above conditions and is recognized by a DFA with exactly 21 states.

Exercise 6.1.10. Add another exercise or two.

Appendix A. Consequences of Linear Algebra

The following matertial is not required in CPSC 421 this year.

If L is a language over an alphabet Σ, we set

$$\text{Count}_L(k) \stackrel{\text{def}}{=} \left|L \cap \Sigma^k\right|,$$

which counts how many words of length k over Σ lie in L. Theorems in linear algebra show that $\text{Count}_L(k)$ must satisfy certain conditions if L is regular and accepted by a DFA with n states.

For example, if

$$\text{Count}_L(k) = \big(C + o(1)\big)10^k/k \tag{6}$$

where C is a positive real number, then facts from linear algebra show that L is not regular; similarly if 10 above is replaced with any positive real.

As an application, since the number of primes less than N is $N/\log N + o(N)$ (this is called the Prime Number Theorem), the language PRIMES of primes written in base 10 is asymptotically

$$\big(C + o(k)\big)10^k/k$$

for some $C > 0$ (the constant C depends on whether or not leading 0's are allowed). It follows that PRIMES is not regular.

Let us briefly describe more general consequences of linear algebra. We call these consequences "linear algebra tests."

A DFA with n states has an *adjacency matrix*, which is an $n \times n$ matrix whose i, j entry counts the number of symbols (in the alphabet, Σ, of the DFA) that take you from state i to state j in the DFA. It follows a DFA has adjacency matrix, M, and recognizes the langauge, L, then

$$\text{Count}_L(k) \stackrel{\text{def}}{=} \left|L \cap \Sigma^k\right|$$

is a sum of the $1, j$ components of M^k over all final states j, were state 1 is the initial state. The Cayley-Hamilton theorem implies that $\text{Count}_L(k)$ satisfies an n-term recurrence equation

$$\text{Count}_L(k) = c_1\text{Count}_L(k-1) + \cdots + c_n\text{Count}_L(k-n) \tag{7}$$

for all $k \geq n$ for fixed integers $c_1, \ldots, c_n$.

As an application, (7) easily implies that

$$L = \{a^m \mid m \text{ is a perfect square}\}$$

is not regular; it also shows that the language

$$L = \{a^m \mid m \in \mathbb{N},\ m \geq 20\}$$

cannot be recognized by a DFA with 20 states or fewer (which is optimal, since there is a 21 state DFA for this language).

The "Jordan canonical form" therem implies that for any regular language, L, there are complex numbers $\lambda_1, \ldots, \lambda_m$ and polynomials $p_1, \ldots, p_m$ such that for k sufficiently large we have

$$\mathrm{Count}_L(k) = \sum_{i=1}^{m} p_i(k)\lambda_i^k.$$

As a consequence, one can show that if

$$\lambda = \lim_{k\to\infty} \frac{\mathrm{Count}_L(k+1)}{\mathrm{Count}_L(k)}$$

exists, then there is a polynomial p such that

$$\mathrm{Count}_L(k) = p(k)\lambda^k(1 + o(1)). \tag{8}$$

This implies that if

$$\mathrm{Count}_L(k) = \big(C + o(k)\big)10^k/k$$

for some $C > 0$, then L is not regular. So the results regarding (8) generalize those of (6).

If $L = \{0^n 1^n \mid n \in \mathbb{N}\}$ (which is a favourite example in textbooks of a non-regular language), then $\mathrm{Count}_L(k)$ alternates between 0 and 1. You get the same counting function for the regular language

$$\{0^{2n} \mid n \in \mathbb{N}\}.$$

Hence linear algebra tests can fail to provide optimal bounds on DFA's and regularity.

One actually gets more information from linear algebra: for example, in (8), λ must be an *algebraic integer*, and C must be an *algebraic number*.

Department of Computer Science, University of British Columbia, Vancouver, BC V6T 1Z4, CANADA.

E-mail address: jf@cs.ubc.ca

URL: http://www.cs.ubc.ca/~jf