بسم اللّه الرّحمن الرّحیم

دانشگاه صنعتی اصفهان ــ دانشکدهٔ مهندسی برق و کامپیوتر

(نیم‌سال تحصیلی ۴۰۲۱)

نظریهٔ زبان‌ها و ماشین‌ها

حسین فلسفین

Identifying Nonregular Languages

Intuition tells us that a language is regular only if, in processing any string, the information that has to be remembered at any stage is strictly limited. This is true, but has to be shown precisely to be used in any meaningful way. There are several ways in which this can be done.

Thus our intuition can sometimes lead us astray, which is why we need mathematical proofs for certainty. In this session, we show how to prove that certain languages are not regular.

We show how to prove that certain languages cannot be recognized by any finite automaton.

شهود می‌تواند گمراه‌کننده باشد. اثبات رسمی لازم است.

Let's take the language $B = \{0^n1^n | n \geq 0\}$. If we attempt to find a DFA that recognizes B, we discover that the machine seems to need to remember how many 0s have been seen so far as it reads the input. Because the number of 0s isn't limited, the machine will have to keep track of an unlimited number of possibilities. But it cannot do so with any finite number of states.

Doesn't the argument already given prove nonregularity because the number of 0s is unlimited? It does not. Just because the language appears to require unbounded memory doesn't mean that it is necessarily so. It does happen to be true for the language B; but other languages seem to require an unlimited number of possibilities, yet actually they are regular.

شهود می‌تواند گمراه‌کننده باشد. اثبات رسمی لازم است.

For example, consider two languages over the alphabet $\Sigma = \{0, 1\}$:

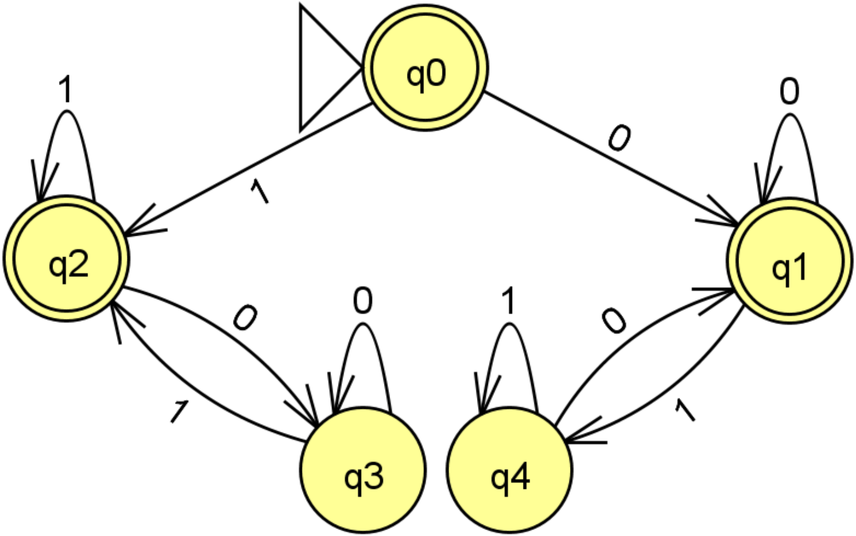
$$C = \{w|w \textit{ has an equal number of } 0\textit{s and } 1\textit{s}\},$$

and

$$D = \{w|w \textit{ has an equal number of occurrences of } 01 \textit{ and } 10 \textit{ as substrings}\}.$$

At first glance, a recognizing machine appears to need to count in each case, and therefore neither language appears to be regular. As expected, C is not regular, but surprisingly D is regular!

DFA نظیر زبان D



The Pumping Lemma for Regular Languages

A proof using the pumping lemma that L cannot be accepted by a finite automaton is a proof by contradiction. In fact, to use the pumping lemma to prove that a language L is not regular, first assume that L is regular in order to obtain a contradiction.

This theorem states that all regular languages have a special property. If we can show that a language does not have this property, we are guaranteed that it is not regular. The property states that all strings in the language can be "pumped" if they are at least as long as a certain special value, called the pumping length. That means each such string contains a section that can be repeated any number of times with the resulting string remaining in the language.

Pumping lemma: *If A is a regular language, then there is a number p (the pumping length) where if s is any string in A of length at least p, then s may be divided into three pieces, $s = xyz$, satisfying the following conditions:*

1. *for each $i \geq 0$, $xy^iz \in A$,*
2. *$|y| > 0$, and*
3. *$|xy| \leq p$.*

Recall the notation where $|s|$ represents the length of string s, y^i means that i copies of y are concatenated together, and y^0 equals ε. When s is divided into xyz, either x or z may be ε, but condition 2 says that $y \neq \varepsilon$. Condition 3 states that the pieces x and y together have length at most p.

بیان دیگری برای لم تزریق

Let A be an infinite regular language. Then there exists some positive integer p such that any $s \in A$ with $|s| \geq p$ can be decomposed as $s = xyz$ with $|xy| \leq p$, and $|y| \geq 1$, such that xy^iz, is also in A for all $i = 0, 1, 2, \ldots$. To paraphrase this, every sufficiently long string in A can be broken into three parts in such a way that an arbitrary number of repetitions of the middle part yields another string in A. We say that the middle string is "pumped," hence the term pumping lemma for this result.

نکتهٔ اساسی و مهم

ما حق تعیین تجزیه را نداریم. یعنی حق نداریم تعیین کنیم که x، y، و z چه هستند یا چه نیستند.

ما صرفاً می‌دانیم که یک چنین تجزیه‌ای (یعنی تجزیه‌ای که سه شرط لم را ارضا می‌کند) وجود دارد. اینکه دقیقاً چیست را نمی‌دانیم. لذا استدلال ما نباید مبتنی بر تعیین تجزیه باشد.

Proof Idea: ***Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA that recognizes A. We assign the pumping length p to be the number of states of M. We show that any string s in A of length at least p may be broken into the three pieces xyz, satisfying our three conditions.***

What if no strings in A are of length at least p? Then our task is even easier because the theorem becomes vacuously true: Obviously the three conditions hold for all strings of length at least p if there aren't any such strings.

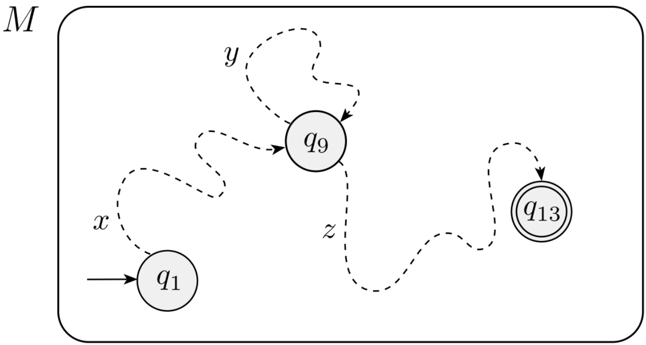
If s in A has length at least p, consider the sequence of states that M goes through when computing with input s. It starts with q_1 the start state, then goes to, say, q_3, then, say, q_{20}, then q_9, and so on, until it reaches the end of s in state q_{13}. With s in A, we know that M accepts s, so q_{13} is an accept state.

If we let $|s| = n$, the sequence of states $q_1, q_3, q_{20}, q_9, \ldots, q_{13}$ has length $n + 1$. Because $n \geq p$, we know that $n + 1$ is greater than p, the number of states of M. Therefore, the sequence must contain a repeated state. This result is an example of the pigeonhole principle.

$$\begin{array}{cccccccccccccccccc}
s = & & s_1 & & s_2 & & s_3 & & s_4 & & s_5 & & s_6 & & \cdots & & s_n & \\
& \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow \\
& q_1 & & q_3 & & q_{20} & & q_9 & & q_{17} & & q_9 & & q_6 & & q_{35} & & q_{13}
\end{array}$$

Example showing state q_9 repeating when M reads s

We now divide s into the three pieces x, y, and z. Piece x is the part of s appearing before q_9, piece y is the part between the two appearances of q_9, and piece z is the remaining part of s, coming after the second occurrence of q_9. So x takes M from the state q_1 to q_9, y takes M from q_9 back to q_9, and z takes M from q_9 to the accept state q_{13}.



Let's see why this division of s satisfies the three conditions.

☞ ***Suppose that we run M on input $xyyz$. We know that x takes M from q_1 to q_9, and then the first y takes it from q_9 back to q_9, as does the second y, and then z takes it to q_{13}. With q_{13} being an accept state, M accepts input $xyyz$. Similarly, it will accept xy^iz for any $i > 0$. For the case $i = 0$, $xy^iz = xz$, which is accepted for similar reasons. That establishes condition 1.***

☞ ***Checking condition 2, we see that $|y| > 0$, as it was the part of s that occurred between two different occurrences of state q_9.***

☞ ***In order to get condition 3, we make sure that q_9 is the first repetition in the sequence. By the pigeonhole principle, the first $p + 1$ states in the sequence must contain a repetition. Therefore, $|xy| \leq p$.***

Proof: Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA recognizing A and $p = |Q|$. Let $s = s_1 s_2 \cdots s_n$ be a string in A of length n, where $n \geq p$. Let $r_1, r_2, \ldots, r_{n+1}$ be the sequence of states that M enters while processing s, so $r_{i+1} = \delta(r_i, s_i)$ for $1 \leq i \leq n$. This sequence has length $n+1$, which is at least $p+1$. Among the first $p+1$ elements in the sequence, two must be the same state, by the pigeonhole principle. We call the first of these r_j and the second r_l. Because r_l occurs among the first $p+1$ places in a sequence starting at r_1, we have $l \leq p+1$. Now let $x = s_1 \cdots s_{j-1}$, $y = s_j \cdots s_{l-1}$, and $z = s_l \cdots s_n$.

* As x takes M from r_1 to r_j, y takes M from r_j to r_j, and z takes M from r_j to r_{n+1}, which is an accept state, M must accept xy^iz for $i \geq 0$.
* We know that $j \neq l$, so $|y| > 0$;
* and $l \leq p+1$, so $|xy| \leq p$.

Thus we have satisfied all conditions of the pumping lemma.

مـــثـــال‌هـــا

Example 1: Let B be the language $\{0^n 1^n | n \geq 0\}$. Assume to the contrary that B is regular. Let p be the pumping length given by the pumping lemma. Choose s to be the string $0^p 1^p$. Because s is a member of B and s has length more than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string $xy^i z$ is in B. Because $|xy| \leq p$ (by condition 3) and the first p symbols of s are 0s (because of the way we chose s), all the symbols in x and y must be 0s. Therefore, $y = 0^k$ for some $k > 0$ (by condition 2). We can get a contradiction from condition 1 by using any number i other than 1, because $xy^i z$ will still have exactly p 1s but will no longer have exactly p 0s. The string $xy^2 z$, for example, is $0^{p+k} 1^p$, obtained by inserting k additional 0s into the first part of s. This is a contradiction, because the pumping lemma says $xy^2 z \in L$, but $p + k \neq p$. This contradicts the pumping lemma and thereby indicates that the assumption that B is regular must be false.

Example 2: *Let $C = \{w|w$ has an equal number of 0s and 1s$\}$. Assume to the contrary that C is regular. Let p be the pumping length given by the pumping lemma, and let s be the string 0^p1^p. With s being a member of C and having length more than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string xy^iz is in C. We would like to show that this outcome is impossible. Because $|xy| \leq p$ (by condition 3), y must consist only of 0s, so $xyyz \notin C$. Therefore, s cannot be pumped. That gives us the desired contradiction.*

Selecting the string s in this example required more care than in the previous example. If we had chosen $s = (01)^p$ instead, we would have run into trouble because we need a string that cannot be pumped and that string can be pumped. Can you see how to pump it? One way to do so sets $x = \varepsilon$, $y = 01$, and $z = (01)^{p-1}$. Then $xy^iz \in C$ for every value of i. If you fail on your first attempt to find a string that cannot be pumped, don't despair. Try another one!

Using Closure Properties

An alternative method of proving that C is nonregular follows from our knowledge that B is nonregular. If C were regular, $C \cap L(0^*1^*)$ also would be regular. The reasons are that the language $L(0^*1^*)$ is regular and that the class of regular languages is closed under intersection. But $C \cap L(0^*1^*)$ equals B, and we know that B is nonregular.

> *Sometimes the easiest way to prove that a language L is not regular is to use the closure theorems for regular languages, either alone or in conjunction with the pumping lemma. The fact that the regular language are closed under intersection is particularly useful.*

***Example 3:* Let $F = \{ww | w \in \{0,1\}^*\}$. We show that F is nonregular, using the pumping lemma. Assume *to the contrary* that F is regular. Let p be the pumping length given by the pumping lemma. Let s be the string 0^p10^p1. Because s is a member of F and s has length more than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, satisfying the three conditions of the lemma. *We show that this outcome is impossible.* Piece y must consist only of 0s, so $xyyz \notin F$.**

Observe that we chose $s = 0^p10^p1$ to be a string that exhibits the "essence" of the nonregularity of F, as opposed to, say, the string 0^p0^p. Even though 0^p0^p is a member of F, it fails to demonstrate a contradiction because it can be pumped.

Example 4: *We show that $L = \{ww^R : w \in \{0,1\}^*\}$ is nonregular, using the pumping lemma. Assume to the contrary that L is regular. Let p be the pumping length given by the pumping lemma. Let s be the string 0^p110^p. Because s is a member of L and s has length more than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, satisfying the three conditions of the lemma. We show that this outcome is impossible. Piece y must consist only of 0s, so $xyyz \notin L$.*

Again, observe that we chose $s = 0^p110^p$ to be a string that exhibits the "essence" of the nonregularity of L, as opposed to, say, the string 0^p0^p. Even though 0^p0^p is a member of L, it fails to demonstrate a contradiction because it can be pumped.

Example 5: Let $D = \{1^{n^2} | n \geq 0\}$. In other words, D contains all strings of 1s whose length is a perfect square. We use the pumping lemma to prove that D is not regular. The proof is by contradiction. Assume to the contrary that D is regular. Let p be the pumping length given by the pumping lemma. Let s be the string 1^{p^2}. Because s is a member of D and s has length at least p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$ the string xy^iz is in D. As in the preceding examples, we show that this outcome is impossible.

Now consider the two strings xyz and xy^2z. These strings differ from each other by a single repetition of y, and consequently their lengths differ by the length of y. By condition 3 of the pumping lemma, $|xy| \leq p$ and thus $|y| \leq p$. We have $|s| = |xyz| = p^2$ and so $|xy^2z| = |xyz| + |y| \leq p^2 + p$. But $p^2 + p < p^2 + 2p + 1 = (p+1)^2$. Moreover, condition 2 implies that y is not the empty string and so $|xy^2z| > p^2$. Therefore, the length of xy^2z lies strictly between *the consecutive perfect squares p^2 and $(p+1)^2$. Hence this length cannot be a perfect square itself:*

$$p^2 = |xyz| < |xy^2z| = p^2 + |y| \leq p^2 + p < p^2 + 2p + 1 = (p+1)^2.$$

So we arrive at the contradiction $xy^2z \notin D$ and conclude that D is not regular.

Example 6: $A = \{a^{2^n} | n \geq 0\}$ ***(Here, a^{2^n} means a string of 2^n a's.) We use the pumping lemma to prove that A is not regular. Assume to the contrary that A is regular. Let p be the pumping length given by the pumping lemma. Choose s to be the string a^{2^p}. Because s is a member of A and s is longer than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, satisfying the three conditions of the pumping lemma. The third condition tells us that $|xy| \leq p$. Furthermore, $p < 2^p$ and so $|y| \leq |xy| \leq p < 2^p$. Therefore, $|xyyz| = |xyz| + |y| < 2^p + 2^p = 2^{p+1}$. The second condition requires $|y| > 0$ so $2^p < |xyyz| < 2^{p+1}$. The length of $xyyz$ cannot be a power of 2. Hence $xyyz$ is not a member of A, a contradiction. Therefore, A is not regular.***

Example 7: ***The language $L = \{w \in \{a,b\}^* : n_a(w) < n_b(w)\}$ is not regular. Let p be the pumping length for L given by the pumping lemma. We pick $s = a^p b^{p+1}$. Now, because $|xy|$ cannot be greater than p, y consists only of as, that is $y = a^k$, $1 \leq k \leq p$. We now pump up, using $i = 2$. The resulting string $a^{p+k}b^{p+1}$ is not in L. Therefore, the pumping lemma is violated, and L is not regular.***

تمرین: با بهره‌گیری از لم تزریق نشان دهید که دو زبان زیر منظم نیستند:

$$L_1 = \{a^n b^m c^n : n \geq 0, m \geq 0\}$$

$$L_2 = \{a^n b^m : n \geq m\}$$

***Example 8:* The language $B = \{0^m 1^n | m \neq n\}$ is not regular.**

روش اول: با بهره‌گیری از خواص بستاری:

Observe that $\overline{B} \cap 0^*1^* = \{0^k 1^k | k \geq 0\}$. If B were regular, then $\overline{B}$ would be regular and so would $\overline{B} \cap 0^*1^*$. But we already know that $\{0^k 1^k | k \geq 0\}$ isn't regular, so B cannot be regular.

روش دوم: با بهره‌گیری از لم تزریق:

Alternatively, we can prove B to be nonregular by using the pumping lemma directly, though doing so is trickier. Assume that $B = \{0^m 1^n | m \neq n\}$ is regular. Let p be the pumping length given by the pumping lemma. Observe that $p!$ is divisible by all integers from 1 to p, where $p! = p(p-1)(p-2)\cdots 1$. The string $s = 0^p 1^{p+p!} \in B$, and $|s| \geq p$. Thus the pumping lemma implies that s can be divided as xyz with $x = 0^a$, $y = 0^b$, and $z = 0^c 1^{p+p!}$, where $b \geq 1$ and $a+b+c = p$. Let s' be the string $xy^{i+1}z$, where $i = p!/b$. Then $y^i = 0^{p!}$ so $y^{i+1} = 0^{b+p!}$, and so $s' = 0^{a+b+c+p!} 1^{p+p!}$. That gives $s' = 0^{p+p!} 1^{p+p!} \notin B$, a contradiction.

ذکر یک نکتهٔ مهم

اگر یک زبان منظم باشد، آنگاه لم تزریق قطعاً برای آن برقرار است. اما اگر لم تزریق برای یک زبان برقرار باشد، آنگاه آن زبان الزاماً منظم نیست. لذا برای اثبات منظم بودن یک زبان، نمی‌توان از برقرار بودن لم تزریق برای آن بهره گرفت. تمرین زیر از کتاب سیپسر را ببینید.

1.54 Consider the language $F = \{\mathtt{a}^i\mathtt{b}^j\mathtt{c}^k \mid i, j, k \geq 0 \text{ and if } i = 1 \text{ then } j = k\}$.

a. Show that F is not regular.

b. Show that F acts like a regular language in the pumping lemma. In other words, give a pumping length p and demonstrate that F satisfies the three conditions of the pumping lemma for this value of p.

c. Explain why parts (a) and (b) do not contradict the pumping lemma.

نکتهٔ مهم: شما حق تعیین x، y، یا z را ندارید، چون با صور وجودی مواجهیم، نه صور عمومی. اثبات شما باید بنحوی باشد که برای هر تجزیهٔ دلخواهی که دو شرط $|xy| \leq p$ و $|y| \geq 1$ را دارا است صادق باشد. مثلاً در اثبات نامنظم بودن زبان

$$L = \left\{ a^m b^n : \frac{m}{n} \text{ is an integer} \right\},$$

بهره‌گیری از رشتهٔ $a^p b^p$ مناسب نیست چون به‌ازای $y = a^p$، تناقض رخ نمی‌دهد (یعنی پامپ کردن میسّر است). شما هم حق ندارید تعیین کنید که y چه هست یا نیست. اما رشتهٔ $s = a^{p+1} b^{p+1}$ مناسب است، چون به‌ازای هر تجزیهٔ دلخواه از s که واجد دو شرط $|xy| \leq p$ و $|y| \geq 1$ باشد داریم $p \leq |y| \leq 1$ و لذا $L \not\ni xy^0z$. (چون در رشته xy^0z، تعداد aها اکیداً کمتر از تعداد bها است و این تعداد، برابر با صفر هم نیست. پس نسبت تعداد aها به bها قطعاً صحیح نیست.)