

Equivalence Properties

For any formal language $L \subseteq \Sigma^*$, the following statements are equivalent:

1. L can be represented by a regular expression.
2. L is regular.
3. L can be recognized by a DFA.
4. L can be recognized by an NFA.
5. L can be recognized by a GNFA.

Properties of Regular Languages

Hantao Zhang

hzhang@cs.uiowa.edu

The University of Iowa, Department of Computer Science

Based on the slides prepared by Professor Rus

Closure Properties

The following operations on formal languages are closed for the collection of all regular languages.

1. (regular operations) union, concatenation, Kleen closure
2. homomorphism, regular substitution
3. complementation, intersection
4. reversal, odd, ...

An Example

Let L be the set of binary strings whose value as binary number is divisible by 5 (assume $(\epsilon) = (000) = 0$). Then L is regular.

- Since there are only five distinct remainders by 5, we may use five states to represent these remainders.
- If $y = x0$, what's the relation between $(y) \bmod 5$ and $(x) \bmod 5$?
- If $y = x1$, what's the relation between $(y) \bmod 5$ and $(x) \bmod 5$?

More on Simulation

- You need to remember a pair of states, one for M_1 and one for M_2 .
- If M_1 has k_1 states and M_2 has k_2 states then you need $k_1 \times k_2$ states to remember simultaneously M_1 and M_2 .
- Transitions of M goes from pair to pair, updating the state for both M_1 and M_2 .
- The start state of M is the pair of start states of M_1 and M_2 ; the accept states of M is the set of pairs containing both accept states of M_1 and M_2 .

Closure under Intersection

- **Proof1:** $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$
- **Proof2:** Because L_1 and L_2 are regular there are automata M_1 and M_2 recognizing L_1 and L_2 , respectively. We construct a finite automaton M from M_1 and M_2 that recognizes $L_1 \cap L_2$ by simulating M_1 and M_2 on any input.
- **Simulation:** As you read the input symbols you keep track of the states that each machine would be in if it had read up to this point.

Note

- $F = F_1 \times F_2: L(M) = L(M_1) \cap L(M_2)$.
- $F = (F_1 \times Q_2) \cup (Q_1 \times F_2): L(M) = ?$

Formal Construction

1. Let M_1 recognize L_1 where $M_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$, and M_2 recognize L_2 where $M_2 = (Q_2, \Sigma, \delta_2, q_0^2, F_2)$
2. We construct M to recognize $L_1 \cap L_2$,
 $M = (Q, \Sigma, \delta, q_0, F)$, where:
 - $Q = \{(r_1, r_2) \mid r_1 \in Q_1, r_2 \in Q_2\}$, i.e., $Q = Q_1 \times Q_2$.
 - Σ is assumed the same for M_1 and M_2 .
 - For each $(r_1, r_2) \in Q$ and $a \in \Sigma$,
 $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$
 - $q_0 = (q_0^1, q_0^2)$
 - $F = F_1 \times F_2$.

Note

- $F = F_1 \times F_2: L(M) = L(M_1) \cap L(M_2).$
- $F = (F_1 \times Q_2) \cup (Q_1 \times F_2): L(M) = L(M_1) \cup L(M_2)$
- $F = (Q_1 - F_1) \times F_2:$
 $L(M) = \overline{L(M_1)} \cap L(M_2) = L(M_2) - L(M_1)$
- $F = F_1 \times (Q_2 - F_2): L(M) = L(M_1) - L(M_2)$
- $F = (Q_1 - F_1) \times (Q_2 - F_2):$
 $L(M) = ?$

Note

- $F = F_1 \times F_2: L(M) = L(M_1) \cap L(M_2).$
- $F = (F_1 \times Q_2) \cup (Q_1 \times F_2): L(M) = L(M_1) \cup L(M_2)$
- $F = (Q_1 - F_1) \times F_2:$
 $L(M) = ?$

Reversal Operation

- For $x = a_1 a_2 \cdots a_n \in \Sigma^*$, define $x^R = a_n \cdots a_2 a_1$. E.g., $100^R = 001$.
- For $L \subseteq \Sigma^*$, define $L^R = \{w^R \mid w \in L\}$.
- **Theorem:** If L is regular, so is L^R .
- **Proof:** Suppose L is recognized by DFA $M = (Q, \Sigma, \delta, q_1, F)$, we construct an NFA $M^R = (Q \cup \{q_0\}, \Sigma, \delta^R, q_0, \{q_1\})$, where $\delta^R(q_0, \epsilon) = F$ and $q \in \delta^R(p, a)$ iff $\delta(q, a) = p$.

Note

- $F = F_1 \times F_2: L(M) = L(M_1) \cap L(M_2).$
- $F = (F_1 \times Q_2) \cup (Q_1 \times F_2): L(M) = L(M_1) \cup L(M_2)$
- $F = (Q_1 - F_1) \times F_2:$
 $L(M) = \overline{L(M_1)} \cap L(M_2) = L(M_2) - L(M_1)$
- $F = F_1 \times (Q_2 - F_2): L(M) = L(M_1) - L(M_2)$
- $F = (Q_1 - F_1) \times (Q_2 - F_2):$
 $L(M) = \overline{L(M_1)} \cap \overline{L(M_2)} = \overline{L(M_1) \cup L(M_2)}$
- ...

Pumping Property

All strings in the language can be “pumped” if they are at least as long as a certain special value, called the **pumping length**

Meaning: each such string in the language contains a section that can be repeated any number of times with the resulting string remaining in the language.

Odd Operation

- For $L \subseteq \Sigma^*$, define $Odd(L) = \{a_1a_3 \cdots a_{2n-1} \mid a_1a_2a_3a_4 \cdots a_{2k-1}a_{2k} \in L\}$.
- **Theorem:** If L is regular, so is $Odd(L)$.
- **Proof:** Suppose L is recognized by DFA $M = (Q, \Sigma, \delta, q_0, F)$, we construct an NFA $M' = (Q, \Sigma, \delta', q_0, F)$, where $p \in \delta'(q, a)$ iff $\delta(q, a) = q'$ and $\delta(q', b) = p$ for some $b \in \Sigma$.

Interpretation

- Recall that $|s|$ represents the length of string s and y^i means that y may be concatenated i times, and $y^0 = \epsilon$
- When $s = xyz$, either x or z may be ϵ , but $y \neq \epsilon$
- Without condition $y \neq \epsilon$ the theorem would be trivially true
- $|xy| \leq p$ is an extra technical condition, useful when proving nonregularity

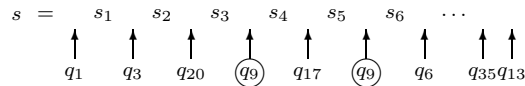
Theorem 1.37

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. $|y| > 0$
3. $|xy| \leq p$

More ideas

- If $s \in A$ and $|s| \geq p$, consider a sequence of states that M goes through to accept s , example: $q_1, q_3, q_{20}, \dots, q_{13}$
- Since M accepts s , q_{13} must be final; if $|s| = n$ then the length of $q_1, q_3, q_{20}, \dots, q_{13}$ is $n + 1$
- Because $|s| = n$ and $|s| \geq p$ it results that $n + 1 > p$.
- By *pigeonhole principle* (if p pigeons are placed into fewer than p holes, some holes must hold more than one pigeon) the sequence $q_1, q_3, q_{20}, \dots, q_{13}$ must contain a repeated state



Proof Idea

Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA that recognizes A

- Assign a pumping length p to be the number of states of M
- Show that any string $s \in A$, $|s| \geq p$ may be broken into three pieces xyz satisfying the pumping lemma's conditions
- If there are no strings in A of length at least p then theorem becomes *vacuously* true because all three conditions hold for all strings of length at least p if *there are no such strings*

Proof

Suppose that we run M on $xyyz$

- **Condition 1:** it is obvious that M accepts xyz , $xyyz$, and in general $xy^i z$ for all $i > 0$. For $i = 0$, $xy^i z = xz$ which is also accepted because z takes M to q_{13}
- **Condition 2:** Since $|s| \geq p$, state q_9 is there. Then because y is the part between two successive occurrences of q_9 , $|y| > 0$.
- **Condition 3:** make sure that q_9 is the first repetition in the sequence. Then by pigeonhole principle, the first $p + 1$ states in the sequence must contain a repetition. Therefore, $|xy| \leq p$

More ideas, continuation

Divide s in to the three pieces: x , y , and z

- Piece x is the part of s appearing before q_9
- Piece y is the part of s between two appearances of q_9
- Piece z is the remaining part of s , after the second appearance of q_9

That is, x takes M from q_1 to q_9 , y takes M from q_9 to q_9 , and z takes M from q_9 to q_{13}

Note

- 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 $j \neq k$, so $|y| > 0$;
- We also know that $k \leq p + 1$, so $|xy| \leq p$

Thus, all conditions are satisfied and lemma is proven

Pumping lemma's proof

Let $M = (Q, \Sigma, \delta, q_1, F)$ be a DFA recognizing A and p be the number of states of M ; let $s = s_1s_2 \dots s_n$ be a string over Σ with $n \geq p$ and r_1, r_2, \dots, r_{n+1} be the sequence of states while processing s , i.e., $r_{i+1} = \delta(r_i, s_i)$, $1 \leq i \leq n$

- $n + 1 \geq p + 1$ and among the first $p + 1$ elements in r_1, r_2, \dots, r_{n+1} two must be the same state, say r_j, r_k .
- Because r_k occurs among the first $p + 1$ places in the sequence starting at r_1 , we have $k \leq p + 1$
- Now let $x = s_1 \dots s_{j-1}$, $y = s_j \dots s_{k-1}$, $z = s_k \dots s_n$.

Observations

- The existence of s contradicts pumping lemma, hence A cannot be regular
- Finding s sometimes takes a bit of creative thinking. Experimentation is suggested

Using pumping lemma

Prove that a language A is not regular using pumping lemma:

1. Assume that A is regular in order to obtain a contradiction
2. The pumping lemma guarantees the existence of a pumping length p such that all strings of length p or greater in A can be pumped
3. Find $s \in A$, $|s| \geq p$, that cannot be pumped: demonstrate that s cannot be pumped by considering all ways of dividing s into x, y, z , showing that for each division one of the pumping lemma conditions, (1) $xy^iz \notin A$, (2) $|y| > 0$, (3) $|xy| \leq p$, fails.

Example, continuation

Consider the cases:

1. y consists of 0s only. In this case $xyyz$ has more 0s than 1s and so it is not in B , violating condition 1
2. y consists of 1s only. This leads to the same contradiction
3. y consists of 0s and 1s. In this case $xyyz$ may have the same number of 0s and 1s but they are out of order with some 1s before some 0s hence it cannot be in B either

The contradiction is unavoidable if we make the assumption that B is regular so B is not regular

Applications

Example 1.38 prove that $B = \{0^n 1^n \mid n \geq 0\}$ is not regular

Assume that B is regular and let p be the pumping length of B . Choose $s = 0^p 1^p \in B$; obviously $|0^p 1^p| > p$. By pumping lemma $s = xyz$ such that for any $i \geq 0$, $xy^i z \in B$

Note

If we take the division $x = z = \epsilon$, $y = 0^p 1^p$ it seems that indeed, no contradiction occurs. However:

- Condition 3 states that $|xy| \leq p$, and in our case $xy = 0^p 1^p$ and $|xy| > p$. Hence, $0^p 1^p$ cannot be pumped
- If $|xy| \leq p$ then y must consist of only 0s, so $xyyz \notin C$ because there are more 0s than 1s. This gives us the desired contradiction

Example 1.39

Prove that $C = \{w \mid w \text{ has an equal number of 0s and 1s}\}$ is not regular

Proof: assume that C is regular and p is its pumping length. Let $s = 0^p 1^p$ with $s \in C$. Then pumping lemma guarantees that $s = xyz$, where $xy^i z \in C$ for any $i \geq 0$.

An alternative method

Use the fact that B is nonregular.

- If C were regular then $C \cap 0^*1^*$ would also be regular because 0^*1^* is regular and intersection of regular languages is a regular language.
- But $C \cap 0^*1^* = B$ which is not regular.
- Hence, C is not regular either.

Other selections

Selecting $s = (01)^p$ leads us to trouble because this string can be pumped by the division: $x = \epsilon$, $y = 01$, $z = (01)^{p-1}$. Then $xy^iz \in C$ for any $i \geq 0$

Note

- Condition 3 is again crucial because without it we could pump s if we let $x = z = \epsilon$, so $xyyz \in F$
- Choosing $s = 0^p10^p1$ is the string that exhibits the essence of the nonregularity of F . If we chose, say $0^p0^p \in F$ we fail because this string can be pumped

Example 1.40

Show that $F = \{ww \mid w \in \{0,1\}^*\}$ is nonregular using pumping lemma

Proof: Assume that F is regular and p is its pumping length. Consider $s = 0^p10^p1 \in F$. Since $|s| > p$ $s = xyz$ and satisfies the conditions of the pumping lemma.

Searching for a contradiction

The elements of D are strings whose lengths are perfect squares. Looking at first perfect squares we observe that they are: 0, 1, 4, 9, 25, 36, 49, 64, 81, ...

- Note the growing gap between these numbers: large members cannot be near each other
- Consider two strings $xy^i z$ and $xy^{i+1} z$ which differ from each other by a single repetition of y .
- If we chose i very large the lengths of $xy^i z$ and $xy^{i+1} z$ cannot be both perfect square because they are too close to each other.

Example 1.41

Show that $D = \{1^{n^2} \mid n \geq 0\}$ is nonregular.

Proof: by contradiction. Assume that D is regular and let p be its pumping length. Consider $s = 1^{p^2} \in D$, $|s| \geq p$. Pumping lemma guarantees that s can be split, $s = xyz$, where for all $i \geq 0$, $xy^i z \in D$

Value of i for contradiction

To calculate the value for i that leads to contradiction we observe that:

- $|y| \leq |s| = p^2$
- Let $i = p^4$. Then
 $|y| \leq p^2 = \sqrt{p^4} < 2\sqrt{p^4} + 1 \leq 2\sqrt{|xy^i z|} + 1$

Turning this idea into a proof

Calculate the value of i that gives us the contradiction.

- If $m = n^2$, calculating the difference we obtain
 $(n+1)^2 - n^2 = 2n + 1 = 2\sqrt{m} + 1$
- By pumping lemma $|xy^i z|$ and $|xy^{i+1} z|$ are both perfect squares. But letting $|xy^i z| = m$ we can see that they cannot be both perfect square if $|y| < 2\sqrt{|xy^i z|} + 1$, because they would be too close together.

Searching for a contradiction

- Let $s = 0^{p+1}1^p$; From decomposition $s = xyz$, from condition 3, $|xy| \leq p$ it results that y consists only of 0s.
- Let us examine $xyyz$ to see if it is in E . Adding an extra-copy of y increases the number of zeros. Since E contains all strings 0^*1^* that have more 0s than 1s will still give a string in E

Example 1.42

Sometimes “pumping down” is useful when we apply pumping lemma.

- We illustrate this using pumping lemma to prove that $E = \{0^i1^j \mid i > j\}$ is not regular
- Proof:** by contradiction using pumping lemma. Assume that E is regular and its pumping length is p .

Nonregular languages

Consider the language $B = \{0^n1^n \mid n \geq 0\}$.

- If we attempt to find a DFA that recognizes B we discover that such a machine needs 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 needs to keep track of an unlimited number of possibilities
- This cannot be done with any finite number of states

Try something else

- Since $xy^iz \in E$ even when $i = 0$, consider $i = 0$ and $xy^0z = xz \in E$.
- This decreases the number of 0s in s .
- Since s has just one more 0 than 1 and xz cannot have more 0s than 1s,
($xyz = 0^{p+1}1^p$ and $|y| \neq 0$)
 xz cannot be in E .

This is the required contradiction

Intuition May Fail Us

• Just because a language appears to require unbounded memory in order to be recognized, it doesn't mean that it is necessarily so

• **Example:**

- $L_1 = \{w \mid w \text{ has an equal number of 0s and 1s}\}$ is not regular
- $L_2 = \{w \mid w \text{ has an equal number of 01 and 10 as substrings}\}$ is regular
- $L_3 = \{xwx \mid w \in \Sigma^*, x \in \Sigma^+\}$ is not regular
- $L_4 = \{xwx \mid w \in \Sigma^*, x \in \Sigma^*\}$ is regular
- $L_5 = \{xx^Rw \mid w \in \Sigma^*, x \in \Sigma^+\}$ is not regular
- $L_6 = \{xwx^R \mid w \in \Sigma^*, x \in \Sigma^+\}$ is regular