Theory of Automata

Shakir Ullah Shah

An Important language

- PALINDROME

  The language consisting of **Λ** and the strings s defined over Σ such that Rev(s) = s. It is to be denoted that the words of PALINDROME are called palindromes.

- Example:

  For Σ={a,b}, PALINDROME={Λ , a, b, aa, bb, aaa, aba, bab, bbb, ...}

PALINDROME

- Sometimes two words in PALINDROME when concatenated will produce a word in PALINDROME
  - **abba** concatenated with **abbaabba** gives **abbaabbaabba** (in PALINDROME)
- But more often, the concatenation is not a word in PALINDROME
  - **aa** concatenated with **aba** gives **aaaba** (NOT in PALINDROME)

Kleene Closure

- **Definition:** Given an alphabet Σ, we define a language in which any string of letters from Σ is a word, even the null string $\mathbf{\Lambda}$. We call this language the **closure** of the alphabet Σ, and denote this language by Σ^*.

- Examples:

  If $\Sigma = \{ 0, 1 \}$ then

  $\Sigma^* = \{ \mathbf{\Lambda}, 0, 1, 00, 01, 10, 11, 000, 001, \dots \}$

  If $\Sigma = \{ a, b \}$ then

  $\Sigma^* = \{ \mathbf{\Lambda}, a, b, aa, ab, ba, bb, aaa, \dots \}$

A subset of $\Sigma *$ is called a language over Σ.

Kleene Closure (contd.)

- Notice that we listed the words in a language in size order (i.e., words of shortest length first), and then listed all the words of the same length alphabetically.
- This ordering is called **lexicographic** order.
- The star in the closure notation is known as the **Kleene star**.
- We can think of the Kleene star as an **operation** that makes, out of an alphabet, an *infinite* language (i.e., *infinitely many* words, each of *finite* length).

Kleene Closure (contd.)

- Canonical:
- Say $\sum = \{0, 1\}$, the string 11 is canonically smaller in $\sum^*$ than the string 000, because 11 is shorter string than 000, or 00 is canonically smaller than 11, because the strings are equal in length but 00 is alphabetically smaller than 11.
- The set $\sum^* = \{\mathbf{\Lambda},$ 0, 1, 00, 01, 10, 11, 000, 001, ...$\}$ is given in its canonical ordering.

Kleene Closure (contd.)

- Example: If S = { aa, b } then

  S* = { **Λ** plus any word composed of factors of aa and b }, or

  S* = { Λ plus any strings of a's and b's in which the a's occur in **even** clumps }, or

  S* = { Λ, b, aa, bb, aab, baa, bbb, aaaa, aabb, baab, bbaa, bbbb, aaaab, aabaa, aabbb, baaaa, baabb, bbaab, bbbaa, bbbbb, … }

  Note that the string aabaaab is not in S* because it has a clump of a's of length 3.

Kleene Closure (contd.)

- Example: Let S = { a, ab }. Then

  S* = { **Λ** plus any word composed of factors of a and ab }, or

  S* = { **Λ** plus all strings of a's and b's except those that start with b and those that contain a double b }, or

  S* = { **Λ**, a, aa, ab, aaa, aab, aba, aaaa, aaab, aaba, abaa, abab, aaaaa, aaaab, aaaba, aabaa, aabab, abaaa, abaab, ababa, … }
- Note that for each word in S*, every b must have an a immediately to its left, so the double b, that is bb, is not possible; neither any string starting with b.

How to prove a certain word is in the closure language S*

- Let S={a,ab} then abaab is in S*, we can factor it as follows:

  abaab = (ab)(a)(ab)

- These three factors are all in the set S, therefore their concatenation is in S*.
- Note that the parentheses, ( ), are used for the sole purpose of demarcating the ends of factors.

How to prove a certain word is in the closure language S*

- Observe that if the alphabet has no letters, then its closure is the language with the null string as its only word; that is

$$\text{if } \sum = \varnothing \text{ (the empty set), then}$$
$$\sum^* = \{ \boldsymbol{\Lambda} \}$$

- Also, observe that if the set S has the null string as its only word, then the closure language S* also has the null string as its only word; that is

$$\text{if } S = \{ \boldsymbol{\Lambda} \},$$
$$\text{then } S^* = \{ \boldsymbol{\Lambda} \}$$

because $\boldsymbol{\Lambda\Lambda} = \boldsymbol{\Lambda}$.

- Hence, the Kleene closure always produces an infinite language unless the underlying set is one of the two cases above.

Positive Closure

- If we wish to modify the concept of closure to refer only the concatenation of **some (not zero)** strings from a set S, we use the notation + instead of *.
- This "plus operation" is called **positive closure**.
- Example: if $\sum = \{ a \}$ then $\sum^{+} = \{ a, aa, aaa, \ldots \}$
- Observe that:

1. If S is a language that **does not** contain **Λ**, then S^{+} is the language S^* without the null word **Λ**.

2. If S is a language that **does** contain **Λ**, then $S^{+} = S^*$

3. Likewise, if $\sum$ is an alphabet, then $\sum^{+}$ is $\sum^*$ without the word **Λ**.

$(S^*)^*$

- What happens if we apply the closure operator twice?
  - We start with a set of words S and form its closure S^*
  - We then start with the set S^* and try to form its closure, which we denote as $(S^*)^*$ or S^{**}
- **Theorem 1:**

  For any set S of strings, we have $S^* = S^{**}$
- Before we prove the theorem, recall from Set Theory that
  - A = B if A is a subset of B **and** B is a subset of A
  - A is a subset of B if for all x in A, x is also in B

Proof of Theorem 1: S* = S**

- Let us first prove that S** is a subset of S*:
  Every word in S** is made up of factors from S*. Every factor from S* is made up of factors from S. Hence, every word from S** is made up of factors from S. Therefore, every word in S** is also a word in S*. This implies that S** is a subset of S*.
- Let us now prove that S* is a subset of S**:
  In general, it is true that for any set A, we have A is a subset of A*, because in A* we can choose as a word any factor from A. So if we consider A to be our set S* then S* is a subset of S**
- Together, these two inclusions prove that S* =

Example for Theorem 1:

- If s={aa, bbb}, then S* is the set of all string where the a's occur in even clumps and b's in the group of 3,6,9.... Some words in S* are aabbbaaaa, bbb, bbbaa
- If we concatenate these three elements of S*, we get one beg word of S** , which is again in S*
- aabbbaaaabbbbbbaa
- [(aa)(bbb)(aa)(aa)][(bbb)][(bbbaa)]

Defining Languages (contd.)

- The languages can be defined in different ways, such as

1. **Descriptive definition**, (covered)
2. Recursive definition,
3. Regular Expressions(RE)
4. Finite Automaton(FA) etc.

- Descriptive Definition:

The language is defined by describing the conditions imposed on its words.

Recursive definition

- **Recursive definition of languages:**

The following three steps are used in recursive definition

1. Some basic words are specified in the language.
2. Rules for constructing more words are defined in the language.
3. No strings except those constructed in above, are allowed to be in the language.

Recursive definition

- **Defining language of INTEGER**
- Step 1: 1 is in **INTEGER**.
- Step 2: If x is in **INTEGER** then x+1 and x-1 are also in **INTEGER**.
- Step 3: No strings except those constructed in above, are allowed to be in **INTEGER**.

Recursive definition

- **Defining language of EVEN**
- Step 1: 2 is in **EVEN**.
- Step 2: If x is in **EVEN** then x+2 and x-2 are also in **EVEN**.
- Step 3: No strings except those constructed in above, are allowed to be in **EVEN**.

Recursive definition

- **Defining the language factorial**
- Step 1: As 0!=1, so 1 is in **factorial**.
- Step 2: n!=n*(n-1)! is in **factorial**.
- Step 3: No strings except those constructed in above, are allowed to be in **factorial**.

Recursive definition

- **Defining the language PALINDROME, defined over** Σ = {a,b}

Step 1:

a and b are in **PALINDROME**

Step 2:

if x is palindrome, then s(x)Rev(s) and xx will also be palindrome, where s belongs to Σ^*

Step 3:

No strings except those constructed in above, are allowed to be in palindrome

Recursive definition

- **Defining the language {$a^n b^n$}, $n=1,2,3,\ldots$, of strings defined over $\Sigma=\{a,b\}$**

Step 1:

ab is in {$a^n b^n$}

Step 2:

if x is in {$a^n b^n$}, then axb is in {$a^n b^n$}

Step 3:

No strings except those constructed in above, are allowed to be in {$a^n b^n$}

Recursive definition

- **Defining the language L, of strings ending in a , defined over $\Sigma=\{a,b\}$**

Step 1:

a is in L

Step 2:

if x is in L then s(x) is also in **L,** where s belongs to Σ^*

Step 3:

No strings except those constructed in above, are allowed to be in **L**

Recursive definition

- **Defining the language L, of strings beginning and ending in same letters , defined over Σ={a, b}**

Step 1:

a and b are in L

Step 2:

(a)s(a) and (b)s(b) are also in **L,** where s belongs to Σ^*

Step 3:

No strings except those constructed in above, are allowed to be in **L**

Recursive definition

- **Defining the language L, of strings containing aa or bb , defined over Σ={a, b}**

Step 1:

aa and bb are in L

Step 2:

s(aa)s and s(bb)s are also in **L,** where s belongs to Σ^*

Step 3:

No strings except those constructed in above, are allowed to be in **L**

Recursive definition

- **Defining the language L, of strings containing exactly aa, defined over $\Sigma=\{a, b\}$**

<u>Step 1:</u>

aa is in L

<u>Step 2:</u>

s(aa)s is also in **L,** where s belongs to b^*

<u>Step 3:</u>

No strings except those constructed in above, are allowed to be in **L**