Chapter 2

Languages

2.1 Introduction

- In English, there are at least three different types of entities: letters, words, sentences.
- letters are from a finite alphabet { a, b, c, . . . , z }
- words are made up of certain combinations of letters from the alphabet. Not all combinations of letters lead to a valid English word.
- sentences are made up of certain combinations of words. Not all combinations of words lead to a valid English sentence.
- So we see that some basic units are combined to make bigger units.
- We want to abstract this to a different level.
- In particular, we will be studying so-called *formal languages*.

2.2 Alphabets, Strings, and Languages

Definition: A *set* is an unordered collection of objects or elements. Sets are written with curly braces {}, and the elements in the set are written within the curly braces.

Examples:

- The set $\{a, b, c\}$ has elements a, b, and c.
- The sets $\{a, b, c\}$ and $\{b, c, b, a, a\}$ are the same since order does not matter in a set and since redundancy does not count.
- The set $\{a\}$ has element a. Note that $\{a\}$ and a are different things; $\{a\}$ is a set with one element a.
- The set $\{x^n : n = 1, 2, 3, \ldots\}$ consists of $x, xx, xxx, \ldots$.
- The set of *even numbers* is $\{0, 2, 4, 6, 8, 10, 12, \ldots\} = \{2n : n = 0, 1, 2, \ldots\}$. In particular, note that 0 is an even number.
- The set of *positive even numbers* is $\{2, 4, 6, 8, 10, 12, \ldots\} = \{2n : n = 1, 2, 3, \ldots\}$.
- The set of *odd numbers* is $\{1, 3, 5, 7, 9, 11, 13, \ldots\} = \{2n + 1 : n = 0, 1, 2, \ldots\}$.

Definition: An *alphabet*, denoted by Σ, is a finite set of fundamental units (called *letters*) out of which we build structure.

Examples:

- The alphabet of lower-case Roman letters is $\Sigma = \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \ldots, \mathrm{z}\}$. (There are 26 lower-case Roman letters.)
- The alphabet of upper-case Roman letters is $\Sigma = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \ldots, \mathrm{Z}\}$. (There are 26 upper-case Roman letters.)
- The alphabet of Arabic numerals is $\Sigma = \{0, 1, 2, \ldots, 9\}$. (There are 10 Arabic numerals.)

Definition: A *string* over an alphabet is a *finite* sequence of letters from the alphabet.

Examples:

- *cat*, $food$, c, and $bbedwxq$ are strings over the alphabet $\Sigma = \{a, b, c, \ldots, z\}$.

- 0173 is a string over the alphabet $\Sigma = \{0, 1, 2, \ldots, 9\}$.

Definition: The *empty string* or *null string*, which we shall denote by Λ, is the string consisting of no letters, no matter what language we're considering.

Definition: Given two strings w_1 and w_2, we define the *concatenation* of w_1 and w_2 to be the string w_1w_2.

Examples:

- If $w_1 = xx$ and $w_2 = x$, then $w_1w_2 = xxx$.
- If $w_1 = abb$ and $w_2 = ab$, then $w_1w_2 = abbab$ and $w_2w_1 = ababb$.
- If $w_1 = \Lambda$ and $w_2 = ab$, then $w_1w_2 = ab$.
- If $w_1 = bb$ and $w_2 = \Lambda$, then $w_1w_2 = bb$.
- If $w_1 = \Lambda$ and $w_2 = \Lambda$, then $w_1w_2 = \Lambda$; i.e., $\Lambda\Lambda = \Lambda$.

Definition: For any string w, we define w^n for $n \geq 0$ inductively as follows:

- $w^0 = \Lambda$;
- $w^{n+1} = w^n w$ for any $n \geq 0$.

Example: If $w = cat$, then $w^0 = \Lambda$, $w^1 = cat$, $w^2 = catcat$, $w^3 = catcatcat$, and so on.

Definition: Given a string s, a *substring* of s is any part of the string s; i.e., w is a substring of s if there exist strings x and y (either or both possibly null) such that $s = xwy$.

Examples:

- Take the string 472828. Then Λ, 282, 4, and 472828 are all substrings of 472828.
- 48 is not a substring of 472828.

Definition: A *formal language* L is a set of strings over an alphabet for which there are explicit rules for the strings in the set. Throughout these notes, we will only consider formal languages, and so we will simplify the discussion by saying *language* instead of *formal language*.

Examples:

- Computer languages, e.g., C or C++ or Java, are formal languages with alphabet $\Sigma =$ { `a`, `b`, ..., `z`, `A`, `B`, ..., `Z`, , `0`, `1`, `2`, ..., `9`, `>`, `<`, `=`, `+`, `-`, `*`, `/`, `(`, `)`, `.`, `,`, `&`, `!`, `%`, `^`, {, }, `|`, `'`, `:`, `;` }. The rules of syntax define the rules for the language.
- The set of valid variable names in C++ is a formal language. What are the alphabet and rules defining valid variable names in C++?

Definition: Those strings that are permissable in the language L are called *words of the language* L.

Remarks:

- A language is just a specific collection of strings.
- We will use the words *string* and *word* interchangeably.
- Thus, for a given string w and a particular language L, we might call w a *word* even if it is not in the language L.

Let us consider some simple examples of languages:

Example: Alphabet $\Sigma = \{x\}$.
Language

$$\begin{aligned} L_0 &= \{\Lambda, x, xx, xxx, xxxx, \ldots\} \\ &= \{x^n \text{ for } n = 0, 1, 2, 3, \ldots\} \\ &= \{x^n : n = 0, 1, 2, 3 \ldots\} \end{aligned}$$

where we interpret x^n to be the string of n x's strung together. In particular, $x^0 = \Lambda$. Note that

- L_0 includes Λ as a word.

- there are different ways we can specify a language.

Example: Alphabet $\Sigma = \{x\}$.
Language

$$\begin{aligned} L_1 &= \{x, xx, xxx, xxxx, \ldots\} \\ &= \{x^n \text{ for } n = 1\ 2\ 3 \ldots\} \\ &= \{x^n : n = 1, 2, 3 \ldots\} \end{aligned}$$

Note that

- L_1 doesn't include Λ as a word.
- there are different ways we can specify a language.

Example: Alphabet $\Sigma = \{x\}$.
Language

$$\begin{aligned} L_2 &= \{x, xxx, xxxxx, xxxxxxx, \ldots\} \\ &= \{x^{\text{odd}}\} \\ &= \{x^{2n+1} : n = 0, 1, 2, 3, \ldots\} \end{aligned}$$

Example: Alphabet $\Sigma = \{0, 1, 2, \ldots, 9\}$.
Language

$$\begin{aligned} L_3 &= \{ \text{ any string of alphabet letters that does not start with the letter "0"} \} \\ &= \{1, 2, 3, \ldots, 9, 10, 11, \ldots\} \end{aligned}$$

Definition: For any set S, we use the notation "$w \in S$" to denote that w is an element of the set S. Also, we use the notation "$y \notin S$" to denote that y is not an element of the set S.

Example: If $L_1 = \{x^n : n = 1, 2, 3 \ldots\}$, then $x \in L_1$ and $xxx \in L_1$, but $\Lambda \notin L_1$.

Definition: The set $\emptyset$, which is called the *empty set*, is the set consisting of no elements.

Fact: Note that $\Lambda \notin \emptyset$ since $\emptyset$ has no elements.

Example: Let $\Sigma = \{a, b\}$, and we can define a language L consisting of all strings that begin with a followed by zero or more b's; i.e.,

$$\begin{aligned} L &= \{a, ab, abb, abbb, \ldots\} \\ &= \{ab^n : n = 0, 1, 2, \ldots\}. \end{aligned}$$

2.3 Set Relations and Operations

Definition: If A and B are sets, then $A \subset B$ (A is a subset of B) if $w \in A$ implies that $w \in B$; i.e., each element of A is also an element of B.

Examples:

- Suppose $A = \{ab, ba\}$ and $B = \{ab, ba, aaa\}$. Then $A \subset B$, but $B \not\subset A$.
- Suppose $A = \{x, xx, xxx, \ldots\}$ and $B = \{\Lambda, x, xx, xxx, \ldots\}$. Then $A \subset B$, but $B \not\subset A$.
- Suppose $A = \{ba, ab\}$ and $B = \{aa, bb\}$. Then $A \not\subset B$ and $B \not\subset A$.

Definition: Let A and B be 2 sets. $A = B$ if $A \subset B$ and $B \subset A$.

Examples:

- Suppose $A = \{ab, ba\}$ and $B = \{ab, ba\}$. Then $A \subset B$ and $B \subset A$, so $A = B$.
- Suppose $A = \{ab, ba\}$ and $B = \{ab, ba, aaa\}$. Then $A \subset B$, but $B \not\subset A$, so $A \neq B$.
- Suppose $A = \{x, xx, xxx, \ldots\}$ and $B = \{x^n : n \geq 1\}$. Then $A \subset B$ and $B \subset A$, so $A = B$.

Definition: Given two sets of strings S and T, we define

$$S + T = \{w : w \in S \text{ or } w \in T\}$$

to be the *union* of S and T; i.e., $S + T$ consists of all words either in S or in T (or in both).

Examples:

- Suppose $S = \{ab, bb\}$ and $T = \{aa, bb, a\}$. Then $S + T = \{ab, bb, aa, a\}$.

Definition: Given two sets S and T of strings, we define

$$S \cap T = \{w : w \in S \text{ and } w \in T\},$$

which is the *intersection* of S and T; i.e., $S \cap T$ consists of strings that are in both S and T.

Definition: Sets S and T are *disjoint* if $S \cap T = \emptyset$.

Examples:

- Suppose $S = \{ab, bb\}$ and $T = \{aa, bb, a\}$. Then $S \cap T = \{bb\}$.
- Suppose $S = \{ab, bb\}$ and $T = \{ab, bb\}$. Then $S \cap T = \{ab, bb\}$.
- Suppose $S = \{ab, bb\}$ and $T = \{aa, ba, a\}$. Then $S \cap T = \emptyset$, so S and T are disjoint.

Definition: For any 2 sets S and T of strings, we define $S - T = \{w : w \in S, w \notin T\}$.

Examples:

- Suppose $S = \{a, b, bb, bbb\}$ and $T = \{a, bb, bab\}$. Then $S - T = \{b, bbb\}$.
- Suppose $S = \{ab, ba\}$ and $T = \{ab, ba\}$. Then $S - T = \emptyset$.

Definition: For any set S, we define $|S|$, which is called the *cardinality* of S, to be the number of elements in S.

Examples:

- Suppose $S = \{ab, bb\}$ and $T = \{a^n : n \geq 1\}$. Then $|S| = 2$ and $|T| = \infty$.

- If $S = \emptyset$, then $|S| = 0$.

Definition: If S is any set, we say that S is *finite* if $|S| < \infty$. If S is not finite, then we say that S is *infinite*.

Examples:

- Suppose $S = \{ab, bb\}$. Then S is finite.
- Suppose $T = \{a^n : n \geq 1\}$. Then T is infinite.

Fact: If S and T are 2 disjoint sets (i.e., $S \cap T = \emptyset$), then $|S + T| = |S| + |T|$.

Fact: If S and T are any 2 sets such that $|S \cap T| < \infty$, then

$$|S + T| = |S| + |T| - |S \cap T|.$$

In particular, if $S \cap T = \emptyset$, then $|S + T| = |S| + |T|$.

Examples:

- Suppose $S = \{ab, bb\}$ and $T = \{aa, bb, a\}$. Then
  - $S + T = \{ab, bb, aa, a\}$
  - $S \cap T = \{bb\}$
  - $|S| = 2$
  - $|T| = 3$
  - $|S \cap T| = 1$
  - $|S + T| = 4$.
- Suppose $S = \{ab, bb\}$ and $T = \{aa, ba, a\}$. Then
  - $S + T = \{ab, bb, aa, ba, a\}$
  - $S \cap T = \emptyset$
  - $|S| = 2$
  - $|T| = 3$
  - $|S \cap T| = 0$
  - $|S + T| = 5$.

Definition: The *Cartesian product* (or *direct* or *cross product*) of two sets A and B is the set $A \times B = \{(x, y) : x \in A, y \in B\}$ of ordered pairs.

Examples:

- If $A = \{ab, ba, bbb\}$ and $B = \{bb, ba\}$, then

  $$A \times B = \{(ab, bb),\ (ab, ba),\ (ba, bb),\ (ba, ba),\ (bbb, bb),\ (bbb, ba)\}.$$

  Note that $(ab, ba) \in A \times B$.

  Also, note that

  $$B \times A = \{(bb, ab),\ (bb, ba),\ (bb, bbb),\ (ba, ab),\ (ba, ba),\ (ba, bbb)\}.$$

  Note that $(bb, ba) \in B \times A$, but $(bb, ba) \notin A \times B$, so $B \times A \neq A \times B$.

We can also define the Cartesian product of more than 2 sets.

Definition: The *Cartesian product* (or *direct* or *cross product*) of n sets $A_1, A_2, \ldots, A_n$ is the set

$$A_1 \times A_2 \times \cdots \times A_n = \{(x_1, x_2, \ldots, x_n) : x_i \in A_i \text{ for } i = 1, 2, \ldots, n\}$$

of ordered n-tuples.

Examples:

- Suppose

  $$\begin{aligned} A_1 &= \{ab, ba, bbb\}, \\ A_2 &= \{a, bb\}, \\ A_3 &= \{ab, b\}. \end{aligned}$$

  Then

  $$\begin{aligned} &A_1 \times A_2 \times A_3 \\ &= \{(ab, a, ab), (ab, a, b), (ab, bb, ab), (ab, bb, b), (ba, a, ab), (ba, a, b), \\ &\quad (ba, bb, ab), (ba, bb, b), (bbb, a, ab), (bbb, a, b), (bbb, bb, ab), (bbb, bb, b)\}. \end{aligned}$$

  Note that $(ab, a, ab) \in A_1 \times A_2 \times A_3$.

Definition: If S and T are sets of strings, we define the *product set* (or *concatenation*) ST to be

$$ST = \{w = w_1 w_2 : w_1 \in S,\ w_2 \in T\}$$

Examples:

- If $S = \{a,\ aa,\ aaa\}$ and $T = \{b,\ bb\}$, then

$$ST = \{ab,\ abb,\ aab,\ aabb,\ aaab,\ aaabb\}$$

- If $S = \{a,\ ab,\ aba\}$ and $T = \{\Lambda,\ b,\ ba\}$, then

$$ST = \{a,\ ab,\ aba,\ abb,\ abba,\ abab,\ ababa\}$$

- If $S = \{\Lambda,\ a,\ aa\}$ and $T = \{\Lambda,\ bb,\ bbbb,\ bbbbbb,\ \ldots\}$, then

$$ST = \{\Lambda,\ a,\ aa,\ bb,\ abb,\ aabb,\ bbbb,\ abbbb,\ \ldots\}$$

Definition: For any set S, define 2^S, which is called the *power set*, to be the set of all possible subsets of S; i.e., $2^S = \{A : A \subset S\}$.

Example: If $S = \{a, bb, ab\}$, then

$$2^S = \{\emptyset, \{a\}, \{bb\}, \{ab\}, \{a, bb\}, \{a, ab\}, \{bb, ab\}, \{a, bb, ab\}\}.$$

Fact: If $|S| < \infty$, then $|2^S| = 2^{|S|}$; i.e., there are $2^{|S|}$ different subsets of S.

2.4 Functions and Operations

Definition: For any string s, the *length* of s is the number of letters in s. We will sometimes denote the length of a string s by length(s) or by $|s|$.

Examples:

- length(cat) = 3. Also, $|cat| = 3$. If we define a string s such that $s = cat$, then $|s| = 3$.

- $|\Lambda| = 0$.

Definition: A *function* (or *operator*, *operation*, *map*, or *mapping*) f maps each element in a domain D into a single element in a range R. We denote this by $f : D \to R$. Also, we say that the mapping f is defined on the domain D and that f is an R-valued mapping. In particular, if the range $R \subset \Re$, i.e., if the range is a subset of the real numbers, then we say that f is a *real-valued* mapping.

Examples:

- Let $\Re$ denote the real numbers, and let $\Re_+$ denote the non-negative real numbers. We can define a function $f : \Re \to \Re_+$ as $f(x) = x^2$.
- If we define f such that $f(3) = 4$ and $f(3) = 8$, then f is *not* a function since it maps 3 to more than one value.
- Let D be any collection of strings, and let R be the non-negative integers. Then we can define $f : D \to R$ to be such that for any string $s \in D$,

  $$f(s) = |s|,$$

  which is the length of s.
- We can define a function $f : \Re \times \Re \to \Re$ to be $f(x, y) = x + y$.
- Let L_1 and L_2 be two sets of strings. Then we can define the *concatenation* operator as the function $f : L_1 \times L_2 \to L_1 L_2$ such that

  $$f(w_1, w_2) = w_1 w_2$$

- Language $L_1 = \{x^n : n \geq 1\}$ from before.
  Can concatenate $a = xxx$ and $b = x$ to get $ab = xxxx$.
  Note that $a, b \in L_1$ and that $ab \in L_1$.
- Language $L_2 = \{x^{2n+1} : n \geq 0\}$ from before.
  Can concatenate $a = xxx$ and $b = x$ to get $ab = xxxx$.
  Note that $a, b \in L_2$ but that $ab \notin L_2$.

Definition: For a mapping f defined on a domain D, we define

$$f(D) = \{f(x) : x \in D\};$$

i.e., $f(D)$ is the set of all possible values that the mapping f can take on when applied to values in D.

Example:

- If $f(x) = x^2$ and $D = \Re$, then $f(D) = \Re_+$, the set of non-negative real numbers.

Definition: Suppose f is a mapping defined on a domain D. We say that D *is closed under mapping* f if $f(D) \subset D$; i.e., if $x \in D$ implies that $f(x) \in D$. In other words, D is closed under f if applying f to any element in D results in an element in D.

Definition: Suppose f is a mapping defined on a domain $D \times D$. We say that D *is closed under mapping* f if $f(D, D) \subset D$; i.e., if $(x, y) \in D \times D$ implies that $f(x, y) \in D$.

Examples:

- $L_1 = \{x^n : n = 1, 2, 3, \ldots\}$ is closed under concatenation.
- $L_2 = \{x^{2n+1} : n = 0, 1, 2, \ldots\}$ is not closed under concatenation since x concatenated with x yields $xx \notin L_2$.

Definition: For any string w, the *reverse* of w, written as reverse(w) or w^R, is the same string of letters written in reverse order. Thus, if $w = w_1 w_2 \cdots w_n$, where each w_i is a letter, then reverse(w) $= w_n w_{n-1} \cdots w_1$.

Examples:

- For $xxxx \in L_1 = \{x^n : n = 1, 2, 3, \ldots\}$, reverse($xxxx$) $= xxxx \in L_1$. We can show that L_1 is closed under reversal.
- Recall L_3 is the set of strings over the alphabet $\Sigma = \{0, 1, 2, \ldots, 9\}$ such that the first letter is not 0. For $48 \in L_3$, reverse(48) $= 84 \in L_3$.
- **Example:** For $90210 \in L_3$, reverse(90210) $= 01209 \notin L_3$. Thus, L_3 is not closed under reversal.

Definition: Over the alphabet $\Sigma = \{a, b\}$, the language **PALINDROME** is defined as

$$\begin{aligned} \textbf{PALINDROME} &= \{\Lambda \text{ and all strings } x \text{ such that reverse}(x) = x\} \\ &= \{\Lambda,\ a,\ b,\ aa,\ bb,\ aaa,\ aba, \ldots\} \end{aligned}$$

Note that for the language **PALINDROME**, the words $abba, a \in$ **PALINDROME**, but their concatenation $abbaa$ is not in **PALINDROME**.

Definition: Suppose $f : D \to \Re$ and $g : D \to \Re$ are real-valued mappings such that $f(x) \leq g(x)$ for all $x \in D$. Then f is a *bounded above* by g, or g is an *upper bound* for f. In addition, if there exists some $x \in D$ such that $f(x) = g(x)$, then we say that g is a *tight* upper bound for f.

Examples:

- If $f(x) = \sin(x)$ and $g(x) = 2$ for all $x \in \Re$, then g is an upper bound of f, but g is not a tight upper bound of f.
- If $f(x) = \sin(x)$ and $g(x) = 1$ for all $x \in \Re$, then g is a tight upper bound of f.
- Suppose $f(x) = x$ and $g(x) = x^2$. Then g is an upper bound for f for all $x \geq 1$, and g is tight since $g(x) = f(x)$ for $x = 1$.
- Suppose $f(x) = x^2$ and $g(x) = 2^x$. Then g is an upper bound for f for all $x \geq 4$. Also, g is a tight upper bound over $x \geq 4$ since $g(x) = f(x)$ for $x = 4$.

2.5 Closures

Definition: Given an alphabet Σ, let Σ^* be the *closure* of the alphabet, which is defined to be the language in which any string of letters from Σ (with possible repetition of letters) is a word in Σ^*, even the null string Λ. This notation is also known as the *Kleene star*. Thus,

$$\Sigma^* = \{w = w_1 w_2 \cdots w_n : n \geq 0,\ w_i \in \Sigma \text{ for } i = 1, 2, \ldots, n\},$$

where we define $w_1 w_2 \cdots w_n = \Lambda$ when $n = 0$.

Example: Alphabet $\Sigma = \{x\}$. Then, the closure of Σ is

$$\Sigma^* = \{\Lambda, x, xx, xxx, \ldots\}$$

Example: Alphabet $\Sigma = \{0, 1, 2, \ldots, 9\}$. Then, the closure of Σ is

$$\Sigma^* = \{\Lambda, 0, 1, 2, \ldots, 9, 00, 01, 02, 03, \ldots\}$$

We can think of the Kleene star as an operation that makes an infinite language (i.e., a language with infinitely many words) out of an alphabet.

Definition: Given a set S of strings, we define S^n, $n \geq 1$, to be

$$\begin{aligned} S^n &= \underbrace{SS\cdots S}_{n \text{ times}} \\ &= \{w = w_1 w_2 \cdots w_n : w_i \in S,\ i = 1, 2, \ldots, n\}. \end{aligned}$$

Note that $S^1 = S$. We also define $S^0 = \{\Lambda\}$.

Example: If $S = \{ab, bbb\}$, then $S^1 = S$, and

$$\begin{aligned} S^2 &= \{abab, abbbb, bbbab, bbbbbb\} \\ S^3 &= \{ababab, ababbbb, abbbbab, abbbbbbb, bbbabab, bbbabbbb, bbbbbbab, bbbbbbbbb\} \end{aligned}$$

We can also apply the star-operator to sets of words:

Definition: If S is a set of words, then S^* is the set of all finite strings formed by concatenating words from S, where any word may be used as often as we like, and where the null string is also included; i.e.,

$$S^* = S^0 + S^1 + S^2 + S^3 + \cdots.$$

In set notation,

$$S^* = \{w = w_1 w_2 w_3 \cdots w_n : n \geq 0 \text{ and } w_i \in S \text{ for all } i = 1, 2, 3, \ldots, n\},$$

where we interpret $w_1 w_2 w_3 \cdots w_n$ for $n = 0$ to be the null string Λ. Thus, $S^0 = \{\Lambda\}$ for any set S. In particular, if $S = \emptyset$, we still have $S^0 = \{\Lambda\}$.

Example: If $S = \{ba,\ a\}$, then

$$\begin{aligned} S^* &= \{\Lambda \text{ plus any word composed of factors of } ba \text{ and } a\} \\ &= \{\Lambda, a, aa, ba, aaa, aba, baa, aaaa, aaba, \ldots\}. \end{aligned}$$

If $w \in S^*$, can bb ever be a substring of w? No.

Proof.

- Suppose xy is a substring of length 2 of w, where x and y are single letters.
- Since $w \in S^*$, we can write $w = w_1 w_2 \cdots w_n$, for some $n \geq 0$, where each $w_i \in S$, $i = 1, 2, \ldots, n$.
- Since $S = \{ba,\ a\}$, there are five possibilities for how the 2-letter substring xy could have arisen:
  1. xy is the concatenation of two 1-letter words from S; i.e., for some $i = 1, 2, \ldots, n-1$, we have that $xy = w_i w_{i+1}$, where w_i and w_{i+1} are words from S having only one letter each. Since the only 1-letter word from S is a, we must have that $w_i = w_{i+1} = a$. In this case, $xy = aa$, which is not bb.
  2. xy is a 2-letter word from S; i.e., for some $i = 1, 2, \ldots, n$, we have that $xy = w_i$, where w_i is a 2-letter word from S. Since the only 2-letter word from S is ba, we must have that $w_i = ba$. In this case, $xy = ba$, which is not bb.
  3. xy is the concatenation of a 1-letter word from S and the first letter of a 2-letter word from S; i.e., for some $i = 1, 2, \ldots, n-1$, we have that $xy = w_i w_{i+1,1}$, where
     - w_i is a 1-letter word from S.
     - w_{i+1} is a 2-letter word of S with $w_{i+1} = w_{i+1,1} w_{i+1,2}$ and $w_{i+1,1}$ and $w_{i+1,2}$ are the two letters of w_{i+1}.

     Since the only 1-letter word from S is a, we must have that $w_i = a$. Since the only 2-letter word from S is ba, we must have that $w_{i+1} = ba$, whose first letter is b. In this case, $xy = ab$, which is not bb.
  4. xy is the concatenation of the second letter of a 2-letter word from S and a 1-letter word from S; i.e., for some $i = 1, 2, \ldots, n-1$, we have that $xy = w_{i,2} w_{i+1}$, where
     - w_i is a 2-letter word of S with $w_i = w_{i,1} w_{i,2}$ and $w_{i,1}$ and $w_{i,2}$ are the two letters of w_i.
     - w_{i+1} is a 1-letter word from S.

     Since the only 2-letter word from S is ba, we must have that $w_i = ba$, whose second letter is a. Since the only 1-letter word from S is a, we must have that $w_{i+1} = a$. In this case, $xy = aa$, which is not bb.

5. xy is the concatenation of the second letter of a 2-letter word from S and the first letter of a 2-letter word from S; i.e., for some $i = 1, 2, \ldots, n-1$, we have that $xy = w_{i,2}w_{i+1,1}$, where
   - w_i is a 2-letter word of S with $w_i = w_{i,1}w_{i,2}$ and $w_{i,1}$ and $w_{i,2}$ are the two letters of w_i.
   - w_{i+1} is a 2-letter word of S with $w_{i+1} = w_{i+1,1}w_{i+1,2}$ and $w_{i+1,1}$ and $w_{i+1,2}$ are the two letters of w_{i+1}.

   Since the only 2-letter word from S is ba, we must have that $w_i = w_{i+1} = ba$, whose first letter is b and whose second letter is a. In this case, $xy = ab$, which is not bb.

- This exhausts all of the possibilities for how a 2-letter substring xy can arise in this example. Since all of them result in $xy \neq bb$, we have completed the proof.

■

Example: If $S = \{xx, xxx\}$, then

$$\begin{aligned} S^* &= \{\Lambda \text{ and all strings of more than one } x\} \\ &= \{\Lambda, xx, xxx, xxxx, \ldots\} \end{aligned}$$

To prove that a certain word is in the closure language S^*, we must show how it can be written as a concatenation of words in S.

Example: If $S = \{ba,\ a\}$, then $aaba \in S^*$ since we can break $aaba$ into the factors $a \in S$, $a \in S$, and $ba \in S$; i.e., $aaba = (a)(a)(ba)$.

Note that there is only one way to do the above factorization into words from S; we then say the factorization is *unique*.

Example: If $S = \{xx, xxx\}$, then $xxxxxx \in S^*$ since $xxxxxx = (xx)(xx)(xx) = (xxx)(xxx)$.

Here, the factorization is not unique.

Example: If $S = \emptyset$, then $S^* = \{\Lambda\}$.

Example: If $S = \{\Lambda\}$, then $S^* = \{\Lambda\}$.

Remarks:

- Two words are considered the same if all their letters are the same and in the same order, so there is only one possible word of no letters, Λ.
- There is an important difference between the word that has no letters Λ and the language that has no words, which we denote by $\emptyset$.
- It is not true that Λ is a word in the language $\emptyset$ since $\emptyset$ doesn't have any words at all.
- If a language L does not contain the word Λ and we wish to add it to L, we use the "union of sets" operation denoted by "+" to form $L + \{\Lambda\}$.
- Note that $L \neq L + \{\Lambda\}$ if $\Lambda \notin L$.
- Note that $L = L + \emptyset$.

Definition: If S is some set of words, then $S^+ = S^1 + S^2 + S^3 + \cdots$, which is the set of all finite strings formed by concatenating some positive number of strings from S.

Example: If $\Sigma = \{x\}$, then $\Sigma^+ = \{x, xx, xxx, \ldots\}$.

Definition: If A and B are sets, then $A \subset B$ (A is a subset of B) if $w \in A$ implies that $w \in B$; i.e., each element of A is also an element of B.

Suppose that we have two sets A and B, and we want to prove that $A = B$. One way of proving this is to show that

1. $A \subset B$, and
2. $B \subset A$.

Example: Suppose $A = \{x, xx\}$ and $B = \{x, xx, xxx\}$. Note that $A \subset B$, but $B \not\subset A$, and so $A \neq B$.

Theorem 1 *For any set S of strings, we have that $S^* = S^{**}$.*

Proof. The way we will prove this is by showing two things:

1. $S^{**} \subset S^*$

2. $S^* \subset S^{**}$

To show part 1, we have to prove that any word w_0 in S^{**} is also in S^*.

- Note that since $w_0 \in S^{**}$, w_0 is made up of factors, say $w_1, w_2, \ldots, w_k$, $k \geq 0$, from S^*; i.e., $w_0 = w_1 w_2 \cdots w_k$, with $k \geq 0$ and $w_i \in S^*$ for $i = 1, 2, \ldots, k$.
- Also, each factor w_i, $i = 1, 2, \ldots, k$, is from S^*, and so it is made up of a nonnegative number of factors from S; i.e., $w_i = w_{i,1} w_{i,2} \cdots w_{i,n_i}$, with $n_i \geq 0$ and $w_{i,j} \in S$ for $j = 1, 2, \ldots, n_i$.
- Therefore, we can write

$$\begin{aligned} w_0 &= w_1 w_2 \cdots w_k \\ &= w_{1,1} w_{1,2} \cdots w_{1,n_1} w_{2,1} w_{2,2} \cdots w_{2,n_2} \cdots w_{k,1} w_{k,2} \cdots w_{k,n_k}, \end{aligned}$$

  where each $w_{i,j} \in S$, $i = 1, 2, \ldots, k$, $j = 1, 2, \ldots, n_i$. So the original word $w_0 \in S^{**}$ is made up of factors from S.
- But S^* is just the language made up of the different factors in S.
- Therefore, $w_0 \in S^*$.
- Since w_0 was arbitrary, we have just shown that every word in S^{**} is also a word in S^*; i.e., $S^{**} \subset S^*$.

To show part 2, note that in general, for any set A, we know that $A \subset A^*$. Hence, letting $A = S^*$, we see that $S^* \subset S^{**}$. ∎