Strings and Languages

"It is always best to start at the beginning"

-- Glynda, the good witch of the North

What is a Language?

- A language is a <u>set</u> of <u>strings</u> made of of symbols from a given <u>alphabet</u>.
- An alphabet is a <u>finite set</u> of <u>symbols</u> (usually denoted by Σ)
  - Examples of alphabets:
    - {0, 1}
    - {α, β, χ, δ, ε, ϕ, γ, η}
    - {a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t , u, v, w, x, y, z}
    - {a}

What is a string?

- A *string over Σ* is a <u>finite sequence</u> (possibly empty) of elements of Σ.
- ε denotes the <u>*null string*</u>, the string with no symbols.
  - Example strings over {a, b}
    - ε, a, aa, bb, aba, abbba
  - NOT strings over {a, b}
    - aaaa…., abca

The length of a string

- The <u>length</u> of a string *x*, denoted |*x*|, is the number of symbols in the string
  - Example:
    - |abbab| = 5
    - |a| = 1
    - |bbbbbbb| = 7
    - | ε | = 0

Strings and languages

- For any alphabet Σ, the <u>set of all strings</u> over Σ is denoted as Σ^*.
- A language over Σ is a subset of Σ^*
  - Example
    - $\{a,b\}^* = \{\varepsilon, a, b, aa, bb, ab, ba, aaa, bbb, baa, \ldots\}$
  - Example Languages over {a,b}
    - {ε, a, b, aa, bb} ∅
    - $\{x \in \{a,b\}^* \mid |x| = 8\}$ $\{x \in \{a,b\}^* \mid |x| \text{ is odd}\}$
    - $\{x \in \{a,b\}^* \mid n_a(x) = n_b(x)\}$ {ε}
    - $\{x \in \{a,b\}^* \mid n_a(x) = 2 \text{ and } x \text{ starts with } b\}$

Concatenation of String

- For $x, y \in \Sigma^*$
  - *xy* is the concatenation of *x* and *y*.
    - x = aba, y = bbb, xy=ababbb
    - For all x, $\varepsilon x = x\varepsilon = x$
  - x^i for an integer i, indicates concatenation of x, i times
    - x = aba, x^3 = abaabaaba
    - For all x, $x^0 = \varepsilon$

Some string related definitions

- x is a substring of y if there exists $w,z \in \Sigma^*$ (possibly ε) such that $y = wxz$.
  - *car* is a substring of *carnage*, *descartes*, *vicar*, *car*, but not a substring of charity.
- x is a suffix of y if there exists $w \in \Sigma^*$ such that $y = wx$.
- x is a prefix of y if there exists $z \in \Sigma^*$ such that $y = xz$.

Operations on Languages

- Since languages are simply sets of strings, regular set operations can be applied:
  - For languages L_1 and L_2 over Σ^*
    - $L_1 \cup L_2$ = all strings in L_1 or L_2
    - $L_1 \cap L_2$ = all strings in both L_1 and L_2
    - $L_1 - L_2$ = strings in L_1 that are not in L_2
    - $L' = \Sigma^* - L$

Concatenation of Languages

- If L_1 and L_2 are languages over Σ^*
  - $L_1L_2 = \{xy \mid x \in L_1 \text{ and } y \in L_2\}$
  - Example:
    - L_1 = {hope, fear}
    - L_2 = {less, fully}
    - L_1L_2 = {hopeless, hopefully, fearless, fearfully}

Concatenation of Languages

- If L is a language over Σ^*
  - L^k is the set of strings formed by concatenating elements of L, k times.
  - Example:
    - L = {aa, bb}
    - L^3 = {aaaaaa, aaaabb, aabbaa, aabbbb, bbbbbb, bbbbaa, bbaabb, bbaaaa}
    - $L^0 = \{\varepsilon\}$

Kleene Star Operation

- The set of strings that can be obtained by concatenating any number of elements of a language L is called the Kleene Star, L^*

[redacted]

- Note that since, L^* contains L^0, ε is an element of L^*

Kleene Star Operation

- The set of strings that can be obtained by concatenating one or more elements of a language L is denoted L^+

[redacted]

Specifying Languages

- How do we specify languages?
  - If language is finite, you can list all of its strings.
    - L = {a, aa, aba, aca}
  - Using basic Language operations
    - $L = \{aa, ab\}^* \cup \{b\}\{bb\}^*$
  - Descriptive:
    - $L = \{x \mid n_a(x) = n_b(x)\}$

Specifying Languages

- Next we will define how to specify languages recursively

- In future classes, we will describe how to specify languages by defining a mechanism for generating the language

- Any questions?

Recursive Definitions

- Definition is given in terms of itself
  - Example (factorial)

[redacted]

$4! = 4 * 3!$

$= 4 * (3 * 2!)$

$= 4 * (3 * (2 * 1!))$

$= 4 * (3 * (2 * (1 * 0!)))$

$= 4 * (3 * (2 * (1 * 1)))$

$= 24$

Recursive Definitions and Languages

- Languages can also be described by using a recursive definition
  1. Initial elements are added to your set (BASIS)
  2. Additional elements are added to your set by applying a rule(s) to the elements already in your set (INDUCTION)
  3. Complete language is obtained by applying step 2 infinitely

Recursive Definitions and Languages

- Example:
  - Recursive definition of Σ^*

1. $\varepsilon \in \Sigma^*$
2. For all $x \in \Sigma^*$ and all $a \in \Sigma$, $xa \in \Sigma^*$
3. Nothing else is in Σ^* unless it can be obtained by a finite number of applications of rules 1 and 2.

Recursive Definitions and Languages

- Let's iterate through the rules for $\Sigma = \{a,b\}$
  - i=0 $\Sigma^* = \{\varepsilon\}$
  - i=1 $\Sigma^* = \{\varepsilon, a, b\}$
  - i=2 $\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb\}$
  - i=3 $\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, bab, bba, bbb\}$
  - …and so on

Recursive Definitions and Languages

- Example:
  - Recursive definition of L^*

1. $\varepsilon \in L^*$
2. For all $x \in L$ and all $y \in L$, $xy \in L^*$
3. Nothing else is in L^* unless it can be obtained by a finite number of applications of rules 1 and 2.

Recursive Definitions and Languages

- Let's iterate through the rules for L = {aa,bb}
  - i=0 $L^* = \{\varepsilon\}$
  - i=1 $L^* = \{\varepsilon, aa, bb\}$
  - i=2 $L^* = \{\varepsilon, aa, bb, aaaa, aabb, bbbb, bbaa\}$
  - i=3 $L^* = \{\varepsilon, aa, bb, aaaa, aabb, bbbb, bbaa, aaaaaa, aaaabb, aabbaa, aabbbb, bbbbaa, bbbbbb, \ldots\}$
  - …and so on

Recursive Definitions – another Example

- Example: Palindromes
  - A palindrome is a string that is the same read left to right or right to left
  - First half of a palindrome is a "mirror image" of the second half
  - Examples:
    - a, b, aba, abba, babbab.

Recursive Definitions – another Example

- Recursive definition for palindromes (pal) over Σ
  1. $\varepsilon \in$ pal
  2. For any $a \in \Sigma$, $a \in$ pal
  3. For any $x \in$ pal and $a \in \Sigma$, $axa \in$ pal
  4. No string is in pal unless it can be obtained by rules 1-3

Recursive Definitions – another Example

- Let's iterate through the rules for pal over $\Sigma = \{a,b\}$
  - i=0 pal = {ε, a, b}
  - i=1 pal = {ε, a, b, aaa, aba, bab, bbb}
  - i=2 pal = {ε, a, b, aaa, aba, bab, bbb, aaaaa, aabaa, ababa, abbba, baaab, ababa, bbabb, bbbbb}

Recursive Definitions – yet another Example

- Example: Fully parenthesized algebraic expressions (AE)
  - $\Sigma = \{ a, (,), +, - \}$
  - All expressions where the parens match correctly are in the language
  - Examples:
    - a, (a + a), (a + (a – a)), ((a + a) – (a + a)), etc.

Recursive Definitions – yet another Example

- Recursive definition for AE
  1. a ∈ AE
  2. For any x, y ∈ AE (x + y) and (x – y) ∈ AE
  3. No string is in pal unless it can be obtained by rules 1-2

Recursive Definitions – yet another Example

- Let's iterate through the rules for AE
  - i=0 AE = {a}
  - i=1 AE = { a, (a+a), (a-a) }
  - i=2 AE = {a, (a+a), (a-a) , (a + (a + a)), (a – (a + a)), (a + (a – a)), (a – (a – a)), ((a + a) + a), ((a + a) – a), …}

Recursive Definitions – a final Example

- $L = \{x \in \{0,1\}^* \mid x = 0^i1^j \text{ and } i \geq j \geq 0\}$
  - In English:
    - strings over the alphabet {0,1}
    - each string contains zero or more 0's followed by a zero or more 1's
    - the number of 1's is greater than or equal to the number of 0's

Recursive Definitions – a final Example

- $L = \{x \in \{0,1\}^* \mid x = 0^i1^j \text{ and } i \geq j \geq 0\}$
- A recursive definition
  1. ε ∈ L
  2. For any $x \in L$, both $0x$ and $0x1 \in L$
  3. No strings are in L unless it can be obtained using rules 1-2.

Later we will prove that this definition does indeed describe L.

Recursive Definitions and Languages

- Questions on Recursive Definition?
- Functions on strings and languages can also be defined recursively.

Structural Induction

- When dealing with languages, it is sometime cumbersome to restate the problems in terms of an integer.
- For languages described using a recursive definition, another type of induction, structural induction, is useful.

Structural Induction

- Principles
  - Suppose
    - U is a set,
    - I is a subset of U (BASIS),
    - Op is a set of operations on U (INDUCTION).
    - L is a subset of U defined recursively as follows:
      - $I \subseteq L$
      - L is closed under each operation in Op
      - L is the smallest set satisfying 1 & 2

Structural Induction

- Then
  - To prove that every element of L has some property P, it is sufficient to show:
    1. Every element of I has property P
    2. The set of elements of L having property P is closed under Op

    #2: If $x \in L$ has property P, Op(x) also must have property P

Structural Induction

- Recall this recursive definition of a language L
  1. $\varepsilon \in L$
  2. For any $x \in L$, both $0x$ and $0x1 \in L$
  3. No strings are in L unless it can be obtained using rules 1-2.

And:

$A = \{x \in \{0,1\}^* \mid x = 0^i1^j \text{ and } i \geq j \geq 0\}$

Show $L \subseteq A$ by structural induction

Structural Induction

- Principles
  - Suppose
    - U is a set $U = \{a,b\}^*$
    - I is a subset of U, $I = \{\varepsilon\}$
    - Op is a set of ops on U. $Op = \{0x, 0x1\}$
    - L is a subset of U defined recursively as follows:
      - $I \subseteq L$
      - L is closed under each operation in Op
      - L is the smallest set satisfying 1 & 2.

Structural Induction

- To prove that every element of L has some property P:
  - Our property is:

    $A = \{x \in \{0,1\}^* \mid x = 0^i1^j \text{ and } i \geq j \geq 0\}$

    P(x) is true if $x \in A$.

Structural Induction

– To prove that every element of L has some property P, it is sufficient to show:
  1. Every element of I has property P
     In our case, must show that ε has property P, I.e. $\varepsilon \in A$, $\varepsilon = 0^i 1^j$, $i \geq j \geq 0$

     Once again, this is the case where i=j=0

Structural Induction

2. The set of elements of L having property P is closed under Op
   If $x \in L$ has property P, Op(x) also must have property P

Assume x has property P,
  $x \in A$, $x = 0^i 1^j$, $i \geq j \geq 0$
  Op1(x) = 0x, which is an element of A
  Op2(x) = 0x1 which is an element of A

Similar proof to induction with no mention of an integer

Structural Induction

- Questions?

Summary

- Languages = set of strings
- Recursive Definition of Languages
- Structural Induction

Questions?

- Any questions?
- Next Time:
  – Our first machine: The Finite Automata!