Université Mohammed Khider-BISKRA

Faculté des Sciences Exactes et la nature et de la vie

Département d'informatique

Cours du Module

Théorie des langages

Pr. CHERIF Foudil

Programme du module Théorie des langages

1. Introduction à la logique formelle
2. Introduction aux langages
3. Typologie des grammaires
4. Les langages réguliers
    a. Les grammaires régulières
    b. Les automates d'états finis
    c. Les expressions régulières
5. Les langages algébriques (contexte libre)
    a. Transformation des grammaires (mot vide, récursivité, ..)
    b. Grammaire de Chomsky
    c. Grammaire de Greibach
    d. Automates à pile
6. Les langages contextuels et les automates à bornes linéaires
7. Les langages de type 0 et les machines de Turring

Motivation du cours

L'objectif de ce cours est une initiation à la théorie des langages formels.

Les langages permettent aux humains d'échanger des informations et des idées et de communiquer avec les machines.

Les langages utilisés entre les humains sont appelés '**langages naturels'**, ils sont généralement **informels et ambigus** et demandent une interprétation par un cerveau humain pour être interprétés correctement.

Les langages créés par l'homme pour communiquer avec la machine sont les langages **formels ou artificiels**. Ils doivent être formalisés et non ambigus pour pouvoir être interprétés par une machine, c'est le but de module.

Chapitre 1: Rappels sur les logiques formelles

1. Définition des systèmes formels

Un système formel est un ensemble de données qui permet de manipuler un ensemble de **symboles** en ne considérant que leur **syntaxe** (structure) sans tenir compte de leur **sémantique**(sens, interprétation).

Un système formel est constitué d'une syntaxe:

- Un alphabet fini de symboles
- Un procédé de construction des formules permettant de décrire les formules bien formées de ce langage.

Exemple : - Un alphabet V= { a, b, c }

•Les formules bien formées: suite de lettres de V contenant une seule fois la lettre a, et une seule fois la lettre c, b se trouve avant c

Chapitre 1: Rappels sur les logiques formelles

2- Introduction à la théorie des langages:

La théorie des langages définit les langages de programmation par contre la compilation transforme les programmes écrits dans ces langages en code machine.

Le programme source est transformé en :

- langage machine absolu (directement exécutable)
- langage machine translatable (nécessite une édition de lien)
- langage d'assemblage (nécessite un assembleur)
- langage évolué (nécessite un compilateur)

La structure de base pour la théorie des langages est la notion de monoïde(est une structure munie d'une loi)

Chapitre 1: Rappels sur les logiques formelles

Un langage sur un ensemble appelé vocabulaire est constitué de caractères ou symboles est un sous ensemble des chaînes finies de caractères.

Un langage est défini par une grammaire.

Les automates sont des machines symboliques validant l'appartenance d'une chaîne donnée au langage qu'il décrive (toutes ces notions seront étudiées dans les chapitres suivants.

- les automates d'états finis
- les automates à pile
- les automates à borne linéaire
- les machines de Turing

3. Structure de monoïde

Un **monoïde** est une structure dont la loi de composition est associative. On appelle **monoïde libre** tout monoïde possédant un élément neutre.

L'exemple qui nous intéresse dans la théorie des langages est l'ensemble des chaînes de caractères finies sur un vocabulaire fini, cet ensemble est muni de l'opération de **concaténation** qui est associative et qui possède un élément neutre, **la chaîne vide**

3.1 Vocabulaire

Un vocabulaire V ou alphabet est un ensemble fini de lettres ou symboles appelés les lettres (lettres, chiffres ou autres sigles)

Exemples:

1) $V = \{ a_1, a2,\ldots,a_n)$ V : l'alphabet a_i : les lettres
2) V= { 1 } alphabet à une lettre
3) V= { 0, 1} alphabet binaire
4) V ={ ., - , / } alphabet morse pour la transmission
5) V = { 0,1,…9,a,b,…z } alphabet quelconque

3.2 Monoïde V^+

On appelle **monoïde** $\mathbf{V}^+$ l'ensemble de toutes les chaînes de longueurs finies non vides définies sur V. Ces chaînes sont appelées **mots** et l'ensemble V^+ est infini.

Autrement dit V^+ est l'ensemble des mots de longueur supérieure ou égale à 1 que l'on peut construire à partir de l'alphabet V

Exemple:

$V = \{ a, b \}$

$V^+ = \{ a,b,aa,bb,ab,ba,bb,aaa,\ldots \}$

3.3 Opération de concaténation

L'opération de concaténation consiste à juxtaposer deux mots afin d'obtenir un nouveau mot. C'est une opération **associative** mais **non commutative**

$x, y \in V^+$ x et y sont deux mots

$x = x_1.x_2\ldots x_k$ / $x_i \in V$

$y = y_1.y_2\ldots.y_p$ / $yi \in V$

$x.y = x_1.x_2\ldots x_k\ y_1.y_2\ldots.y_p$

$(x.y).z = x.(y.z)$. est associative

$x.y \neq y.x$. n'est pas commutative

3.4 Monoïde libre V^*

L'opération de concaténation admet un élément neutre qui est la **chaîne vide** de longueur nulle et notée par **ε**

$x.\varepsilon = \varepsilon.x = x$

Donc on peut définir $V^* = V^+ \cup \{\varepsilon\}$

3.5 Longueur d'un mot

La longueur d'un mot x que l'on note généralement |x| fait correspondre à chaque mot le nombre de symboles qui le composent.

On définit un mot particulier appelé **mot vide**, ce mot n'est composé d'aucun caractère, sa longueur est donc nulle ($|\varepsilon| = 0$).

3.5 Sous mot

On dit que y Є V^* est un sous mot (ou facteur) de x Є V^* s'il existe des mots finis u, v Є V^* tel que x = u y v et |y| ≤ |x|

Si x = y v on dit que y est facteur gauche ou préfixe de x

Si x= v y on dit que y est facteur droit ou suffixe de x

Exemple

V= { a, b }

x = abbbbaa et y = bbb donc x = ab**y**aa sous mot

x = bbbaa et y = bbb donc x = yaa facteur gauche