Examen TLA

Grammaire

Grammaire est un quadruplet : (T,V,S,P) :
T : Terminaux (Min)
V : Variables : Non Terminaux (Maj)
S : Start Symbol (axiome)
P : Production Rules

Séquence Alpha générée de Beta aprés l'application de N>0 : A => + B
Séquence Alpha générée de Beta aprés l'application de N>=0 : A => * B

Protophrase : Chaine derivable a partir de l'axiome S

Une grammaire définit un seul langage
Un meme langage peut etre engendré par plusieurs grammaires

Dérivation la plus a gauche / la plus à droite

Arbre syntaxique :
- Racine : Axiome
- Feuilles : Terminaux
- Noeux : Non Terminaux

Grammaire ambigue => Existe une chaine admettant deux arbres de dérivation différents

Grammaire Réguliére (type 3) :
- A gauche un seul non terminal
- A droite : (2 symboles)
    - Terminal.NonTerminal
    - Terminal
    - Epsilon

Grammaire Hors Contexte (type 2):
- A gauche un seul non terminal
- A droite : combinaison de terminaux et non terminaux

Grammaire contextuelle (type 1) : La tete est une chaine

Type 3 inc Type2 inc Type1(contextulle) inc Type0(generale)

Propriétés de GHC :

- Fermeture par union/concatenation/*/+/image miroir
- Non fermeture par intersection/complémentation

Forme Normale de Chomsky :

- **Coté droit des régles :**
  - **Deux symbols non terminaux**
  - **Un symbole terminal**

Convert to chomsky normal form:

- **Remove epsilon productions : https://www.youtube.com/watch?v=mlXYQ8ug2v4&list=PLBInK6fEyqRgp46KUv4ZY69yXmpwKOlev&index=78 (Si l'axiome a une production epsilon est apprait dans une partie droite on ajoute S' => S | epsilon**
- **Replace every terminal with a variable A->a B->b….**
- **Make sure every production have exactly 2 variables**
- **Remove unit productions A-> B : https://www.youtube.com/watch?v=B2o75KpzfU4&list=PLBInK6fEyqRgp46KUv4ZY69yXmpwKOlev&index=77**

Eliminer recursivité à gauche :

- **Remove epsilon productions**
- **Remove cycles S->A A->S**
- **On choisit un ordre de non terminaux**
  - **On les traite 1 a 1 et on regarde les recursivité immédiates**

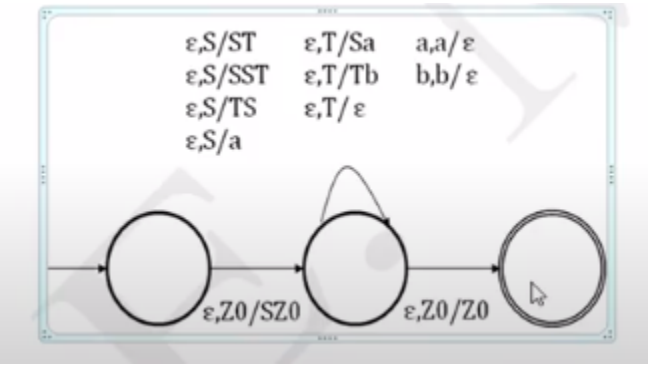
Grammaire Generant L(G)* a partir de G:

- **On ajoute un axiome S' -> SS' | epsilon sur G**

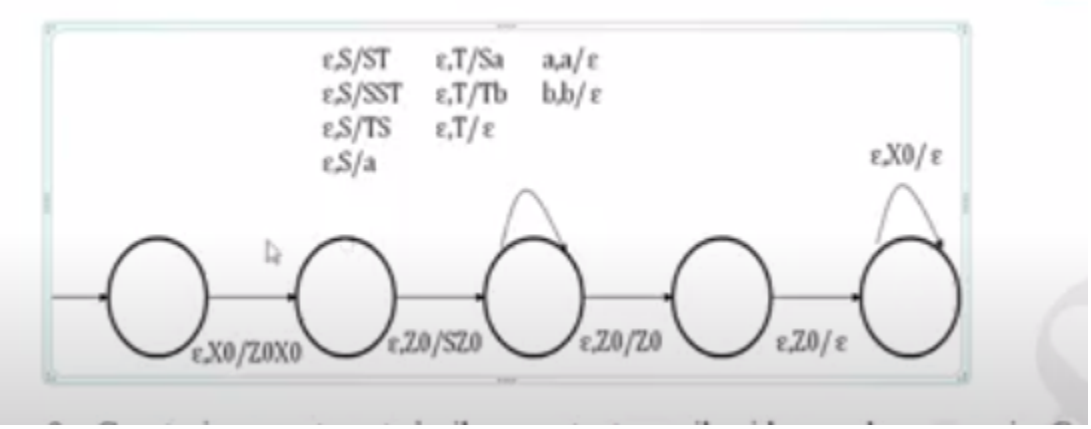
Grammaire Generant L(G1) union L(G2):

- **S'il existe des noms communs, on procéde au renommage (dans les productions)**
- **On ajoute un axiome Sunion -> S(1) | S(2)**

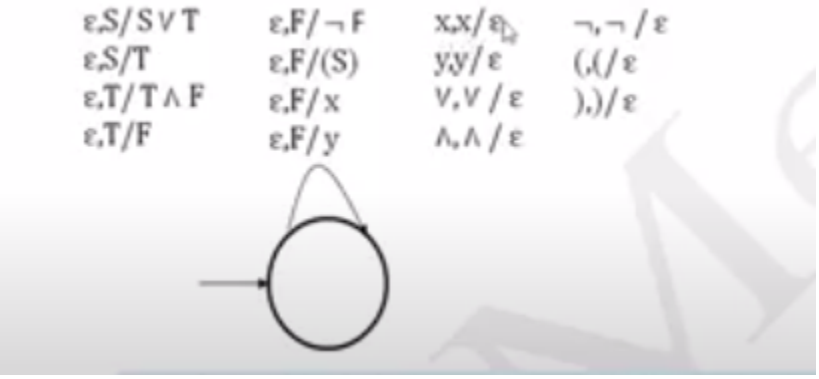
Generer un automate a pile acceptant **par etat final** un grammaire :

- Etat initial / etat centre / etat final
- 

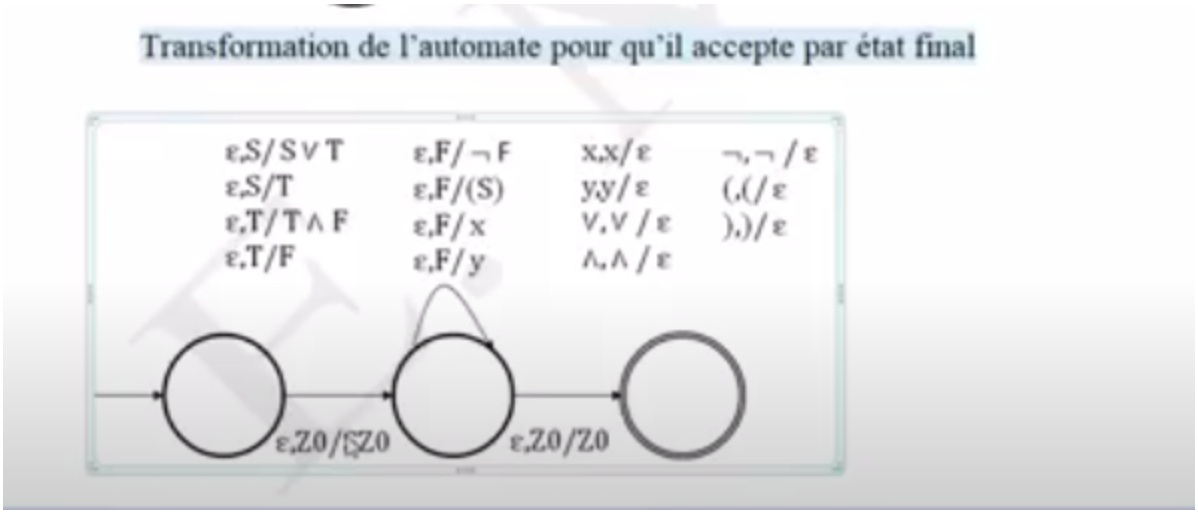
Generer un automate a pile acceptant **par pile vide** un grammaire à partir de etat final :

- 

Generer un automate a pile acceptant **par pile vide**:

- 

Le convertir en par etat final :



Pour chaque L(G) tel que epsilon n'appartient pas à L(G) et G est GHC il existe une grammaire C sous forme normale de chomsky tel que L(G) = L(C)
Une grammaire est dite propre si elle ne posséde pas de régles epsilon ou de régles A->B

Mise sous forme normale de chomsky
https://www.youtube.com/watch?v=FNPSlnj3Vt0&list=PLBlnK6fEyqRgp46KUv4ZY69yXmpwKOIev&index=79

Analyse Lexicale

Analyse décendante (Top down) :

- Dérivations les plus à gauches

Analyse SLR et LL(1)

- Elimination de la récursivté et de la factorisation

Montrer que grammaire est LL(1) :

- Non recursive + non ambigue + factorisé

2. S'il existe une production $A \rightarrow \alpha \mid \beta$ avec ($\alpha \neq \beta$) alors
   i. Il n'existe pas de terminal **a** tel que **α** et **β** génèrent tous deux des chaînes qui commencent par **a** **(PREMIER(α) ∩ PREMIER(β) = Ø)**
   ii. Au plus un de **α** et **β** peut générer la chaîne **ε** **(PREMIER(α) ∩ PREMIER(β) = Ø)**
   iii. Si $\beta \Rightarrow^* \varepsilon$ alors **α** ne dérive aucune chaîne commençant par un terminal qui est dans **SUIVANT(A)**. **($\varepsilon \in$ PREMIER(β) ET PREMIER(α) ∩ SUIVANT(A) = Ø)**

-