Ambiguïté

Nous allons dans ce qui suit définir l'ambiguïté par rapport à la notion de dérivation gauche, ou dérivation droite ou arbre de dérivation (**cours sur les langages algébrique envoyé le 23 Novembre 2018**).

1. Dérivation

Lorsque nous avons introduit au début du cours la notion de grammaire (dans le cas général), nous avons défini la notion de dérivation qui permet de générer des mots à partir d'une grammaire.

Exemple 1.

Soit G<{a, b}, {S}, P, S> la grammaire où P est défini comme suit :

P = {S $\rightarrow$ a S b S / b S a S /ε }

Pour montrer que w = ababba $\in$ L(G), nous allons utiliser les dérivations pour générer ce mot à partir de l'axiome S comme suit :

S|-- a**S**bS |-- abSaSb**S** |-- abSaSbb**S**aS |-- ab**S**aSbbaS |-- abaSbba**S** |-- aba**S**bba |-- ababba

A chaque étape nous avons remplacé un S (parmi un ensemble), sans suivre un ordre particulier, par une de ses productions. Pour la première dérivation, nous avons remplacé S par aSbS. Pour la 2ème dérivation, nous avons choisi de remplacer le premier S (en rouge) par sa production bSaS. Au niveau de la 3ème dérivation, nous avons remplacé le dernier S par bSaS et ainsi de suite jusqu'à obtenir le mot ababba. Il n'y avait pas un ordre précis de remplacement.

Maintenant, nous allons organiser nos dérivations en définissant un ordre pour le remplacement d'une variable par une de ses productions. Il ya deux stratégies. La première est ***la dérivation à gauche*** (A chaque étape, on remplace la variable la plus à gauche). La deuxième est la dérivation à droite (à chaque étape, on remplace la variable la plus à droite).

Définition d'une dérivation Gauche. Soit G<X,V,P,S> une grammaire algébrique. On dit qu'un mot w s'obtient par dérivation gauche s'il est généré à partir de l'axiome en dérivant toujours le non terminal le plus à gauche.

Si nous reprenons l'exemple 1, nous obtenons le mot ababba en faisant les dérivations à gauche de la manière suivante :

S|-- a**S**bS|-- ab**S**aSbS |-- aba**S**bS |-- abab**S** |-- ababb**S**aS |-- ababba**S** |-- ababba

Définition d'une dérivation droite. Soit G<X,V,P,S> une grammaire algébrique. On dit qu'un mot w s'obtient par dérivation droite s'il est généré à partir de l'axiome en dérivant toujours le non terminal le plus à droite.

Si nous reprenons la grammaire de l'exemple 1, nous obtenons le mot ababba en faisant les dérivations à droite de la manière suivante :

S|-- aSbS|-- aSbbSaS |-- aSbbSa |-- aSbba|-- abSaSbba|-- abSabba |-- ababba

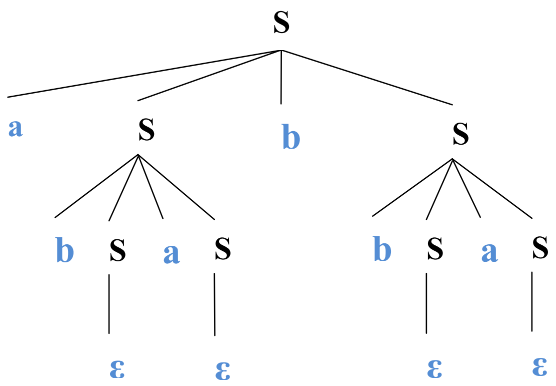
Nous allons citer un théorème qui nous dit qu'à toute dérivation droite, il existe une dérivation gauche équivalente.

Nous pouvons également générer un mot en utilisant la notion d'arbre de dérivation qui est définie ci-dessous :

Définition d'un arbre de dérivation pour un mot. Soit G<X, V, P, S> une grammaire à contexte libre. Ar est un arbre de dérivation pour G si et seulement si :

- Chaque nœud de Ar est étiqueté d'un symbole appartenant à $X \cup V \cup \{\varepsilon\}$
- La racine de l'arbre est étiquetée S (axiome)
- Si un nœud interne est étiqueté A alors $A \in V$
- Si un nœud n est étiqueté A et ses fils $n_1, n_2, \ldots, n_n$ sont étiquetés respectivement $X_1, X_2, \ldots X_n$, alors $A \rightarrow X_1 X_2 \ldots X_n$ est une production de P
- Si un nœud est étiqueté ε, alors c'est le seul fils de son père.

Si nous revenons à l'exemple 1, l'arbre de dérivation pour le mot w = ababba est le suivant :



2. *Ambiguïté*

Définition 1. Un mot w est ambigu s'il existe, pour ce mot, plus d'un arbre de dérivation (resp. plus d'une dérivation droite, resp. plus d'une dérivation gauche).

Définition 2. Une grammaire algébrique G<X,V,P,S> est ambigüe s'il existe un $w \in L(G)$ tel que w est ambigu.

Si nous reprenons l'exemple 1, nous pouvons voir que le mot w = ababba est ambigu. Pour montrer cela il suffit de donner deux dérivations gauches pour ce mot, ou deux dérivations droites ou deux arbres de dérivations.

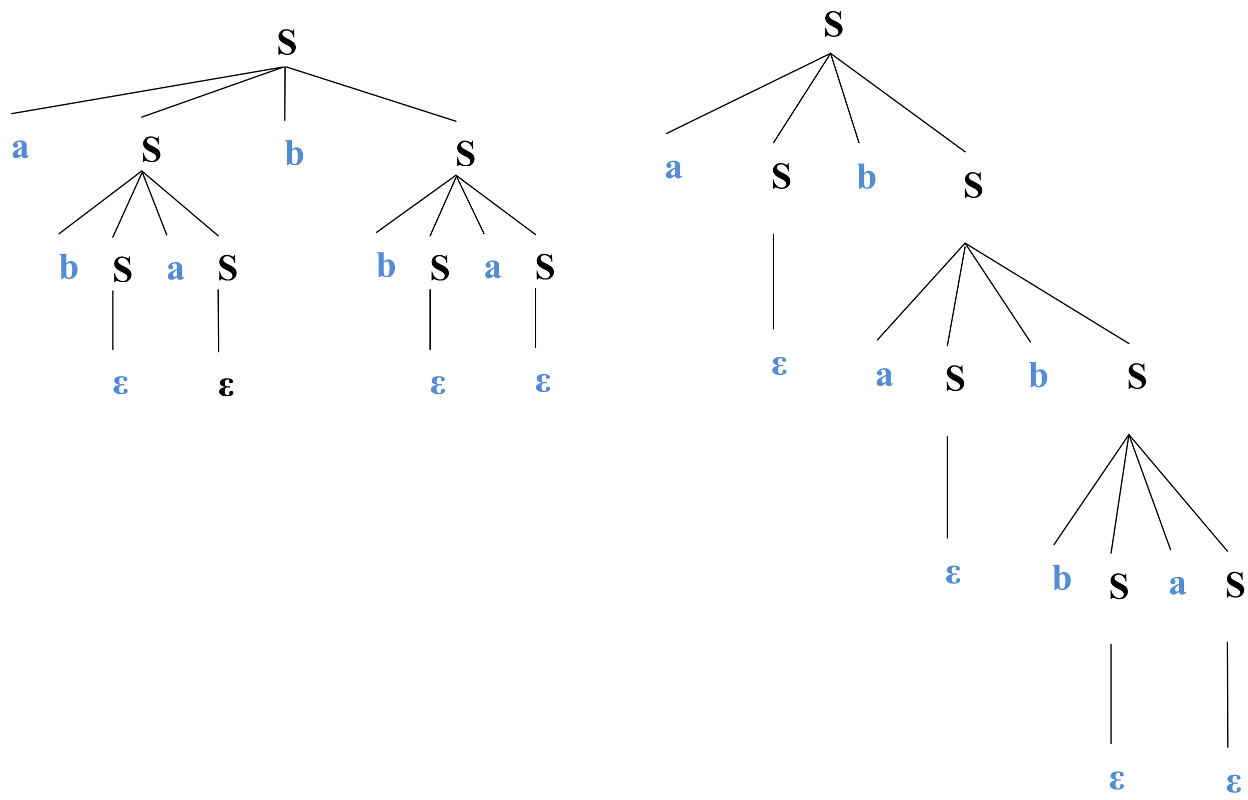
- *Voici deux dérivations gauche pour le mot w*

1. S|-- aSbS|-- abS |-- abaSbS |-- « ababS |-- ababbSaS |-- ababbaS |-- ababba
2. S|-- aSbS|-- abSaSbS |-- abaSbS |-- ababS |-- ababbSaS |-- ababbaS |-- ababba

On va dire donc que le mot w ababba est ambigu et par conséquent la grammaire G est ambiguë.

- Nous donnons également deux dérivations droites pour le mot w
1. S|-- aSbS|-- aSbbSaS |-- abaSbS |-- « ababS |-- ababbSaS |-- ababbaS |-- ababba
2. S|-- aSbS|-- aSbbSaS |-- aSbbSa |-- aSbba|-- abSaSbba|-- abSabba |-- ababba

- Les deux arbres de dérivations pour le mot sont



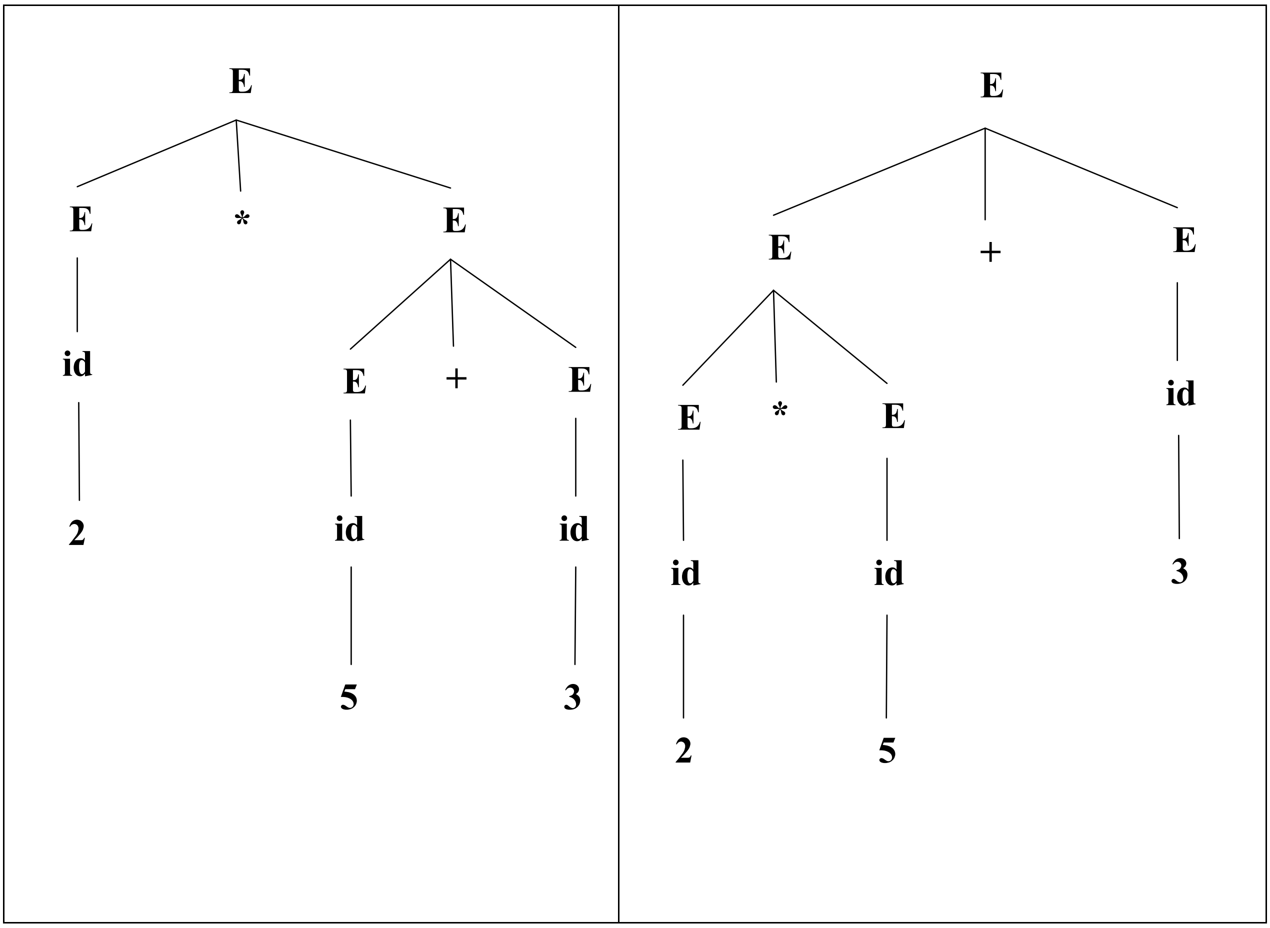
***Définition 3*.** Un langage algébrique a une ambiguïté inhérente si toutes les grammaires qui l'engendrent sont ambiguës.

Exemple 2.

Soit G<{a, b}, {S}, P, E> la grammaire où P est défini comme suit :

P = {E → E + E / E *E/ id
Id → 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9}

Montrons que cette grammaire est ambigüe. Pour cela il suffit de trouver un mot w ∈ L(G) pour lequel il existe au moins deux arbres de dérivations :



Pour éliminer cette ambiguïté, il faut ré-écrire (trouver) une nouvelle grammaire qui permette de générer des expressions arithmétiques (ici nous avons utilisé seulement + et *) en prenant en considération la priorité des opérateurs et en introduisant les parenthèses. Cette grammaire peut être la suivante :

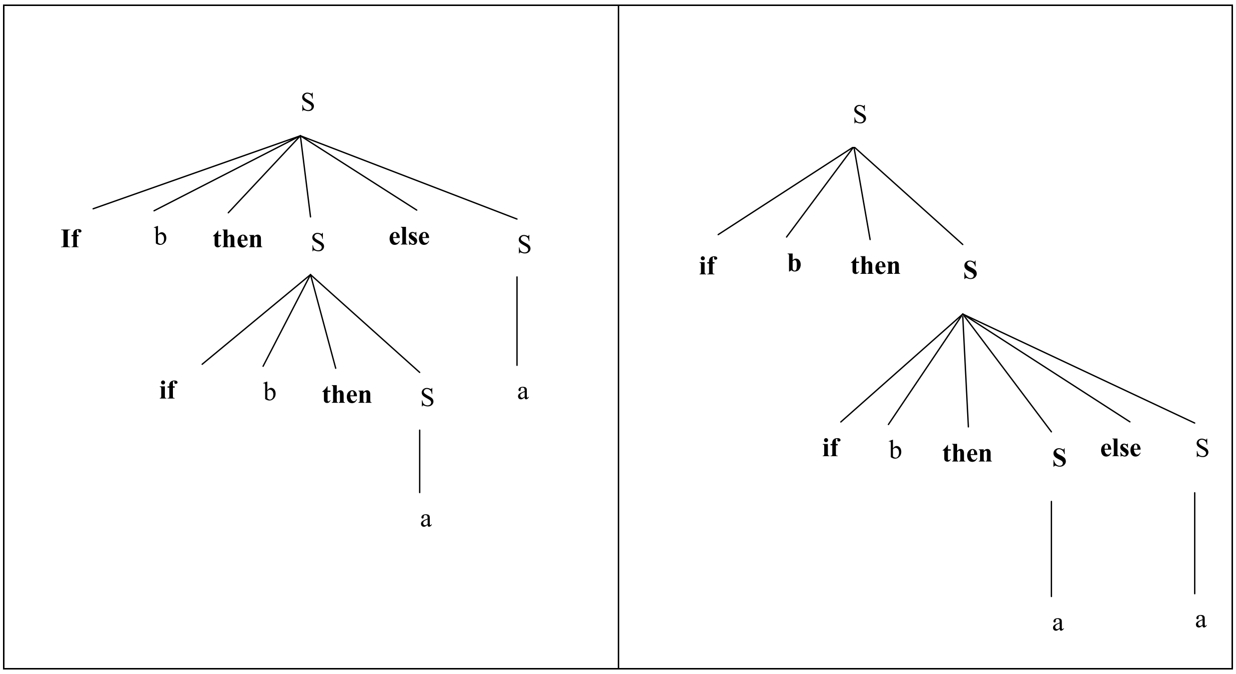
G<{0, .., 9}, {E, T, F}, P, E> la grammaire où P est défini comme suit :

P = {E → E + T / T
T → T * F / F
F → (E) / id
Id → 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9}

Exemple 3 Soit G la grammaire sont les règles de productions sont définies comme suit :

P = {S → **if** b **then** S **else** S / **if** b **then** S / a }

Les deux arbres de dérivations suivants pour le mot w = **if** b **then** **if** b **then** a **else** a montrent que la grammaire est ambigüe.



Dans la grammaire de l'exemple 3, la partie else peut être associée soit au premier If ou au second.

Pour éliminer cette ambigüité, il faut retravailler la grammaire et séparer les cas. La nouvelle grammaire est donnée dans ce qui suit :

P = {$S_1 \rightarrow$ **if** b **then** S_1 / **if** b **then** S_2 **else** S_1 / a
$S_2 \rightarrow$ **if** b **then** S_2 else S_2 / a }