# TP RdF, semaine 12: chaînes, langages et grammaires

François Lepan

11 avril 2013

## 1 Distance de chaînes

### 1.1 À la main

| | | e | x | c | u | s | e | d |
|---|---|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| e | 1 | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| x | 2 | 1 | 0 | 1 | 2 | 3 | 4 | 5 |
| h | 3 | 2 | 1 | 1 | 2 | 3 | 4 | 5 |
| a | 4 | 3 | 2 | 2 | 2 | 3 | 4 | 5 |
| u | 5 | 4 | 3 | 3 | 2 | 3 | 4 | 5 |
| s | 6 | 5 | 4 | 4 | 3 | 2 | 3 | 4 |
| t | 7 | 6 | 5 | 5 | 4 | 3 | 3 | 4 |
| e | 8 | 7 | 6 | 6 | 5 | 4 | 3 | 4 |
| d | 9 | 8 | 7 | 7 | 6 | 5 | 4 | 3 |

### 1.2 Sur machine

**Pour 'abacc'**

```
?- levenshtein('abacc','aabbc',D).
D = 3.
?- levenshtein('abacc','ababcc',D).       Distance moyenne = 2.
D = 1.
?- levenshtein('abacc','babbcc',D).
D = 2.

?- levenshtein('abacc','bccba',D).
D = 4.
?- levenshtein('abacc','bbbca',D).         Distance moyenne = 4
D = 3.
?- levenshtein('abacc','cbbaaaa',D).
D = 5.

?- levenshtein('abacc','caaaa',D).
D = 4.
?- levenshtein('abacc','cbcaab',D).       Distance moyenne = 3.6
```

```
D = 4.
?- levenshtein('abacc','baaca',D).
D = 3.
```

Donc 'abacc' appartient à la classe 1.

**Pour 'ccab'**

```
?- levenshtein('ccab','aabbc',D).
D = 4.
?- levenshtein('ccab','ababcc',D).        Distance moyenne = 4.33
D = 4.
?- levenshtein('ccab','babbcc',D).
D = 5.

?- levenshtein('ccab','bccba',D).
D = 3.
?- levenshtein('ccab','bbbca',D).          Distance moyenne = 4
D = 4.
?- levenshtein('ccab','cbbaaaa',D).
D = 5.

?- levenshtein('ccab','caaaa',D).
D = 3.
?- levenshtein('ccab','cbcaab',D).        Distance moyenne = 3
D = 2.
?- levenshtein('ccab','baaca',D).
D = 4.
```

Donc 'ccab' appartient à la classe 3.

**Pour 'ccbba'**

```
?- levenshtein('ccbba','aabbc',D).
D = 3.
?- levenshtein('ccbba','ababcc',D).       Distance moyenne = 4
D = 5.
?- levenshtein('ccbba','babbcc',D).
D = 4.

?- levenshtein('ccbba','bccba',D).
D = 2.
?- levenshtein('ccbba','bbbca',D).      Distance moyenne = 3
D = 3.
?- levenshtein('ccbba','cbbaaaa',D).
D = 4.

?- levenshtein('ccbba','caaaa',D).
D = 3.
?- levenshtein('ccbba','cbcaab',D).       Distance moyenne = 3.66
```

```
D = 4.
?- levenshtein('ccbba','baaca',D).
D = 4.
```

Donc 'ccbba' appartient à la classe 2

**Pour 'bbaaac'**

```
?- levenshtein('bbaaac','aabbc',D).
D = 4.
?- levenshtein('bbaaac','ababcc',D).      Distance moyenne = 3.66
D = 3.
?- levenshtein('bbaaac','babbcc',D).
D = 4.

?- levenshtein('bbaaac','bccba',D).
D = 4.
?- levenshtein('bbaaac','bbbca',D).        Distance moyenne = 3
D = 3.
?- levenshtein('bbaaac','cbbaaaa',D).
D = 2.

?- levenshtein('bbaaac','caaaa',D).
D = 3.
?- levenshtein('bbaaac','cbcaab',D).       Distance moyenne = 3
D = 3.
?- levenshtein('bbaaac','baaca',D).
D = 3.
```

Donc 'bbaaac' peut être de classe 2 ou de classe 3

# 2 Arbre de dérivation pour une grammaire

## 2.1 A la main : la grammaire G

- Alphabet A = {a,b,c}
- Axiome = S
- Non-terminaux = {A,B}
- Règles de production P =

```
S --> cAb
A --> aBa
B --> aBa
B --> cb
```

### 2.1.1 De quel type est cette grammaire ?

C'est une grammaire algébrique car elle est de la forme $R_i : T \rightarrow x$ où x est terminal ou non.

### 2.2 Montrez que cette grammaire génère le langage $\mathbf{L(G) = \{}ca^ncba^nb \mid n \geqslant 1\mathbf{\}}$

On peut voir sur la Fig. 1 que à chaque fois on rajoute autant de a de chaque coté du cb centrale (cas d'arrêt).

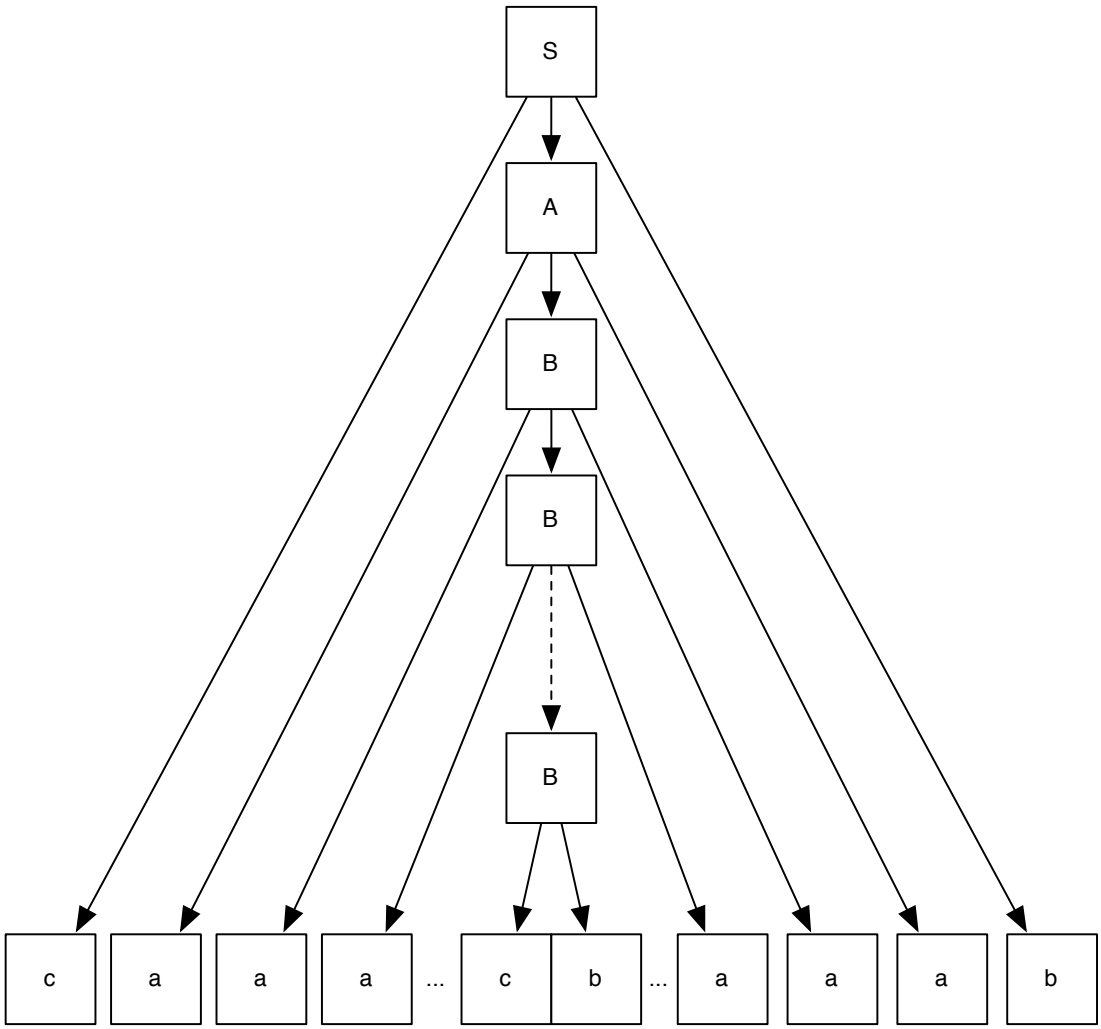


FIGURE 1 – Arbre de dérivation de la grammaire G

### 2.3 Arbres de dérivation

## 3 Palindromes

### 3.1 Grammaire

```
W := aWa | bWb | cWc | dWd | eWe | fWf | gWg | hWh | iWi | jWj | kWk | lWl | mWm | nWn
W := oWo | pWp | qWq | rWr | sWs | tWt | uWu | vWv | wWw | xWx | yWy | zWz |
W := a | b | c | d | e | f | g | h | i | j | k | l | m | n
W := o | p | q | r | s | t | u | v | w | x | y | z |
W := epsilon
```

Cette grammaire est une grammaire algébrique.

### 3.2 Arbres de dérivation

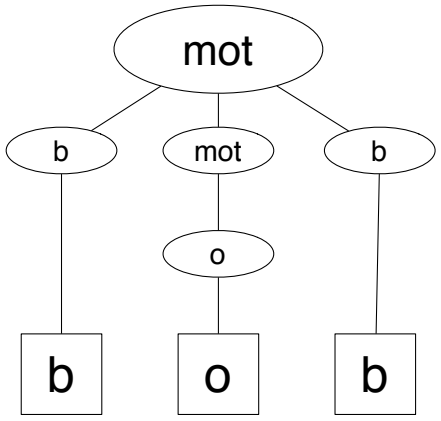


FIGURE 2 – Dérivation du mot 'bob'

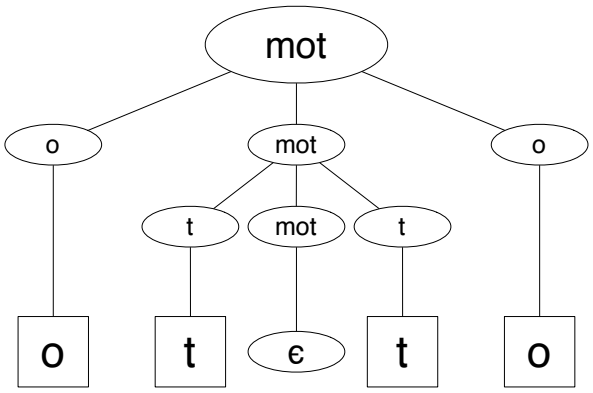


FIGURE 3 – Dérivation du mot 'otto'

### 3.3 Implémentation en Prolog

Nous avons deux implémentations en PROLOG, l'une basée sur l'inversion d'atome, l'autre sur l'extraction de sous atomes.

```
atom_reverse(X,X) :- atom_length(X,0).
atom_reverse(X,X) :- atom_length(X,1).
atom_reverse(X,Y) :-
        atom_concat(A,B,X),
        atom_length(A,1),
        atom_reverse(B,B2),
        atom_concat(B2, A,Y).

palindrome(X) :- atom_reverse(X, X).

palindrome2(X) :- atom_length(X,0).
palindrome2(X) :- atom_length(X,1).
palindrome2(X) :-
        atom_length(X,S),
        P is S-1,
        L is P-1,
        sub_atom(X,0,1,_,A),
        sub_atom(X,P,1,_,A),
        sub_atom(X,1,L,_,W),
        palindrome2(W).
```