

# Corpus arborés et parsing

Cours 6

**Santiago Herrera**  
[s.herrera@parisnanterre.fr](mailto:s.herrera@parisnanterre.fr)

# Règle Grew de réécriture

shift : changement de direction des arêtes

- shift  $A \Rightarrow B$

Toutes les arêtes qui sortent de A, sont rédigérées pour sortir de B

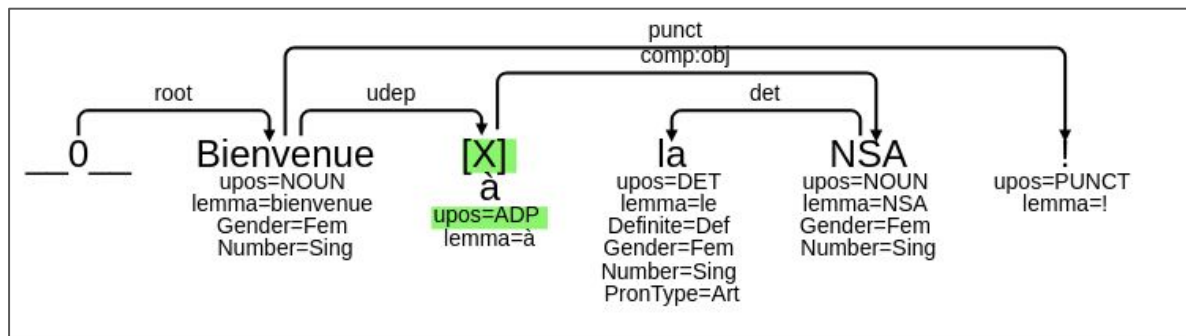
- shift\_out  $B = [\text{subj}|\text{obj}] \Rightarrow C$

On déplace les arêtes avec les relations subj et obj qui sortent de B pour sortir de C

- shift\_in  $C = [\text{suj}|\text{obj}] \Rightarrow D$

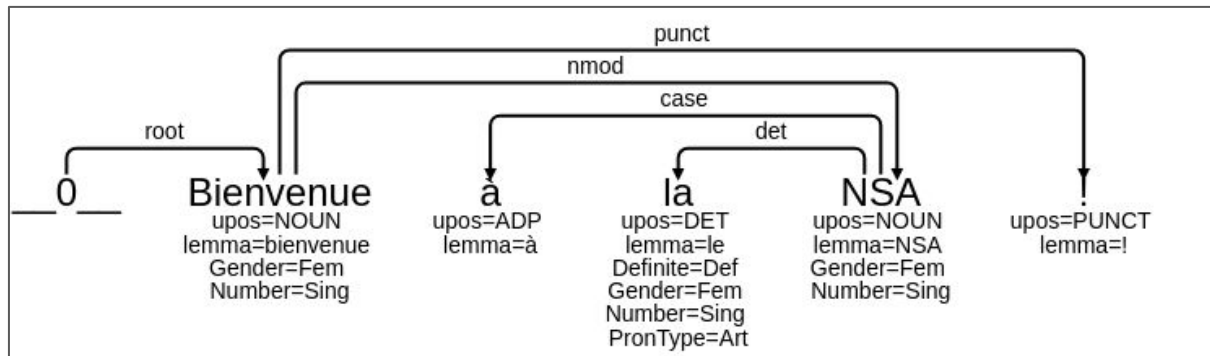
Les arêtes qui arrivent dans C avec les relations subj et obj sont déplacées pour arriver à D

## exo Grew



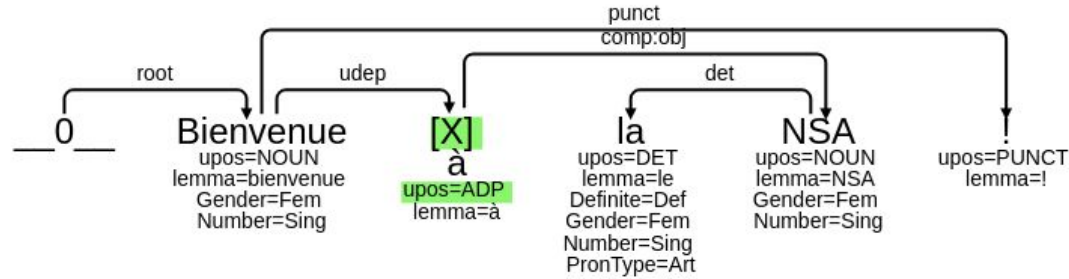
SUD

Convertir l'annotation SUD vers UD : déplacer la relation udep pour qu'elle arrive au nœud 'NSA'.

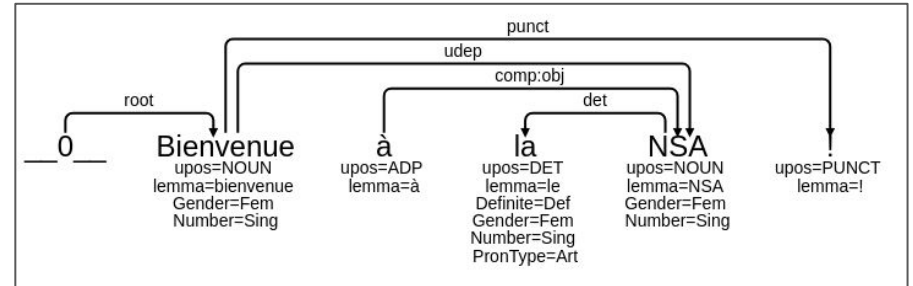


UD

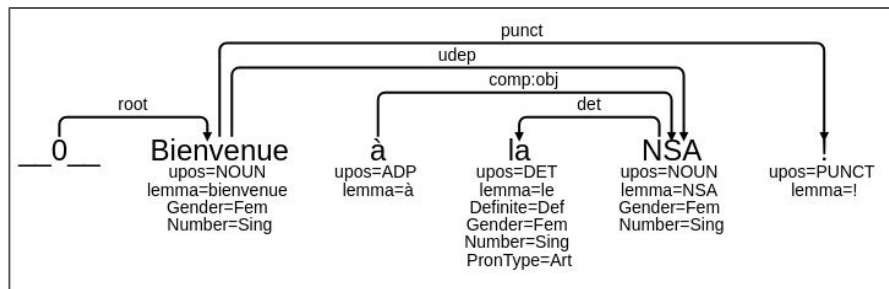
# Correction : Grew



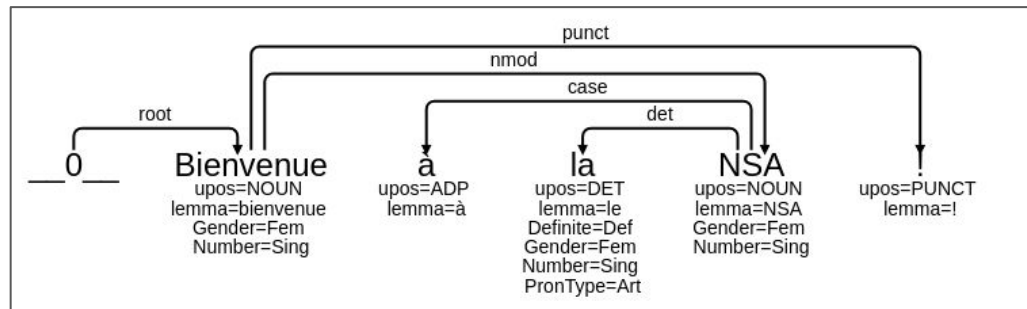
```
1 rule UDversSUD_head {  
2 pattern {X[form="à"]; e:X->Y}  
3 commands {  
4   shift_in X==>Y  
5 }  
6 }
```



# Correction : Grew



```
1 rule UDversSUD_deprel {
2 pattern {e:G-[udep]->X; Y[form="à"]; d:Y->X}
3 commands {
4   del_edge e; add_edge G-[nmod]->X;
5   del_edge d; add_edge X-[case]->Y
6 }
7 }
```

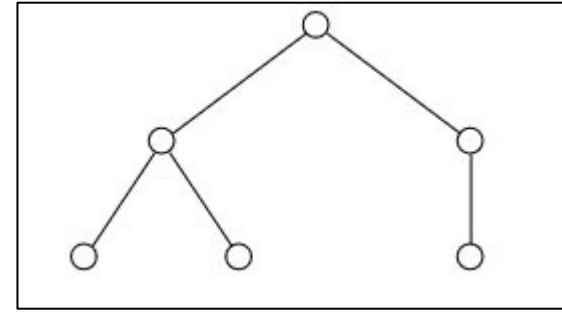
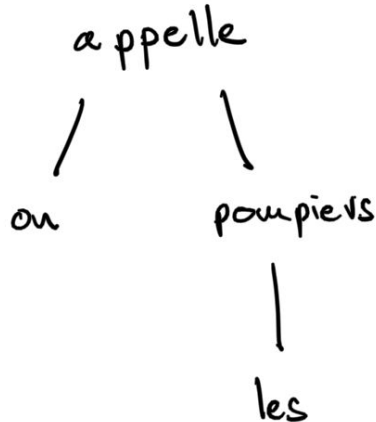


# Arbre de dépendance ordonné

Encodage de deux types d'information

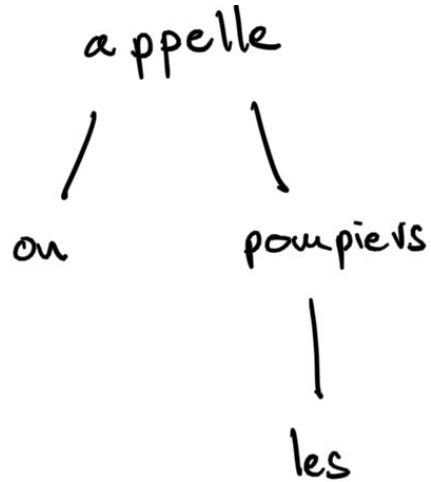
## Structure hiérarchique

- On appelle les pompiers

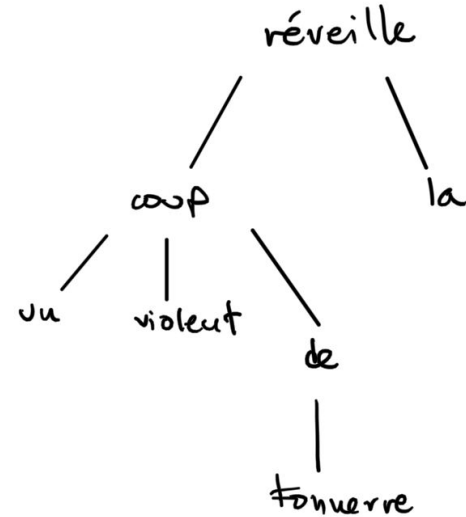


# Projection d'un nœud

- Projection maximale



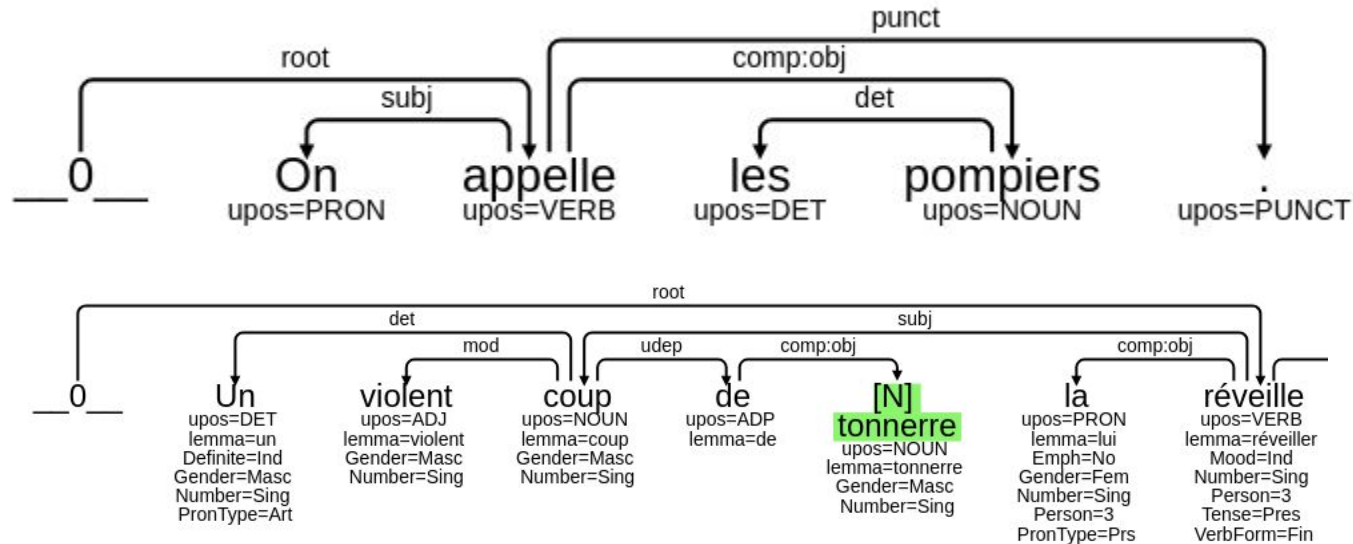
Un violent coup de tonnerre la réveille.



# Arbre de dépendance ordonné

Encodage de deux types d'information

## Structure d'ordre (linéaire)





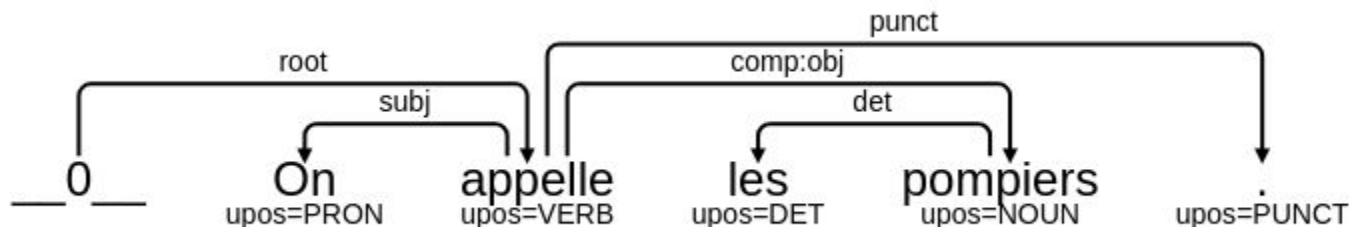
# Projectivité d'un arbre

Une propriété résultante des arbres de dépendance ordonnés

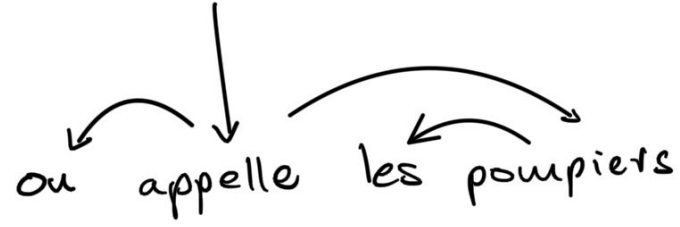
- Un arbre de dépendance est projectif quand “tous les mots se placent autour de leur gouverneur”

Quand chacune de ses projections maximales est continue dans la chaîne linéaire.

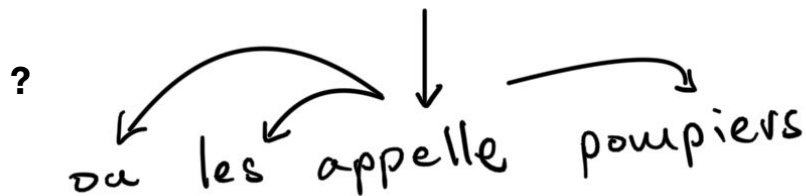
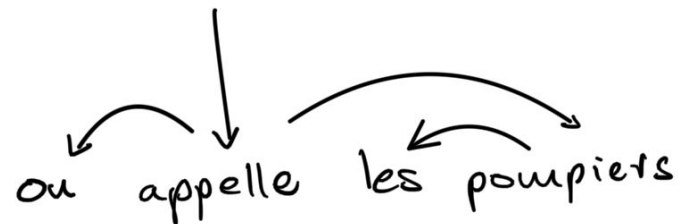
Si et seulement si les dépendances ne se croisent pas dans la représentation en ligne



# Exemples

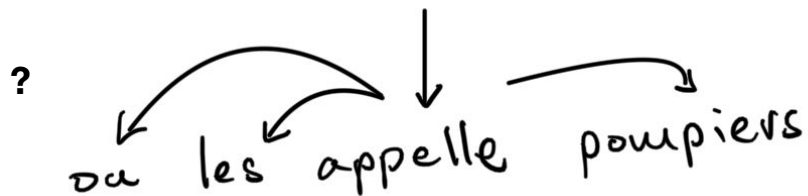
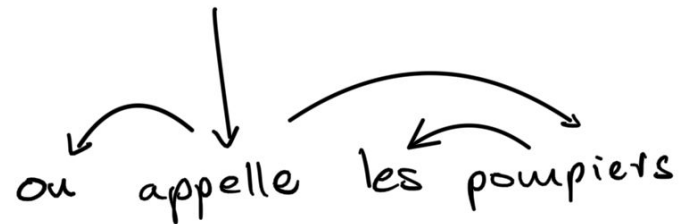


# Exemples

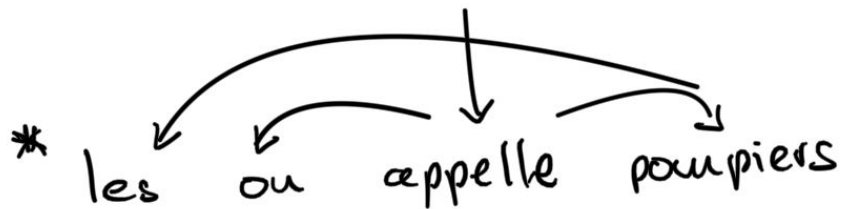


(on les appelle des pompiers)

# Exemples



(on les appelle des pompiers)

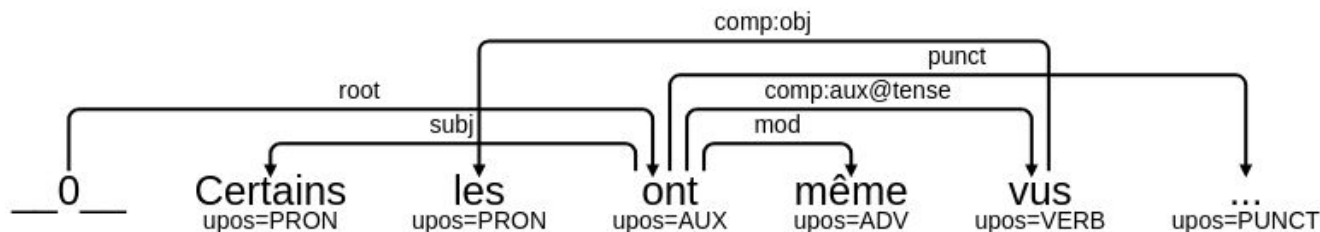


# Des arbres non projectifs

- Un arbre de dépendance est non projectif quand il y a “au moins un mot qui ne se place pas autour de son gouverneur”

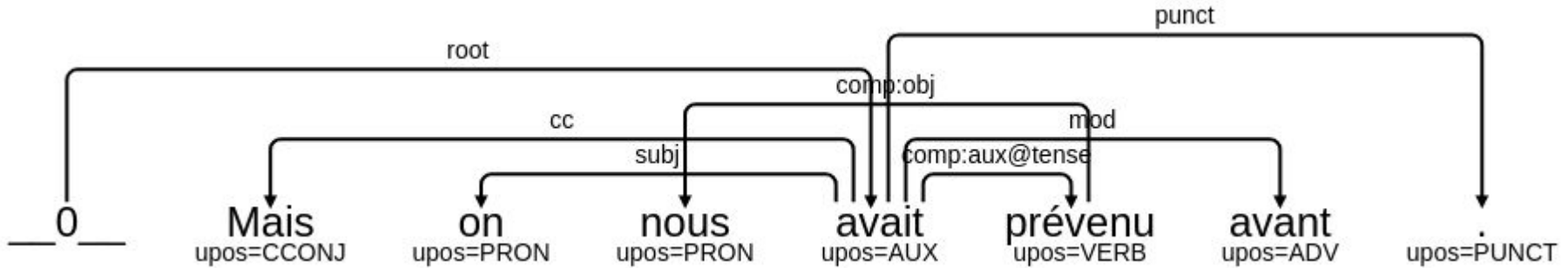
Quand il y a au moins une de ses projections maximales qui n'est pas continue dans la chaîne linéaire.

Si et seulement si les dépendances se croisent dans la représentation en ligne\*



\*Cela concerne aussi la racine

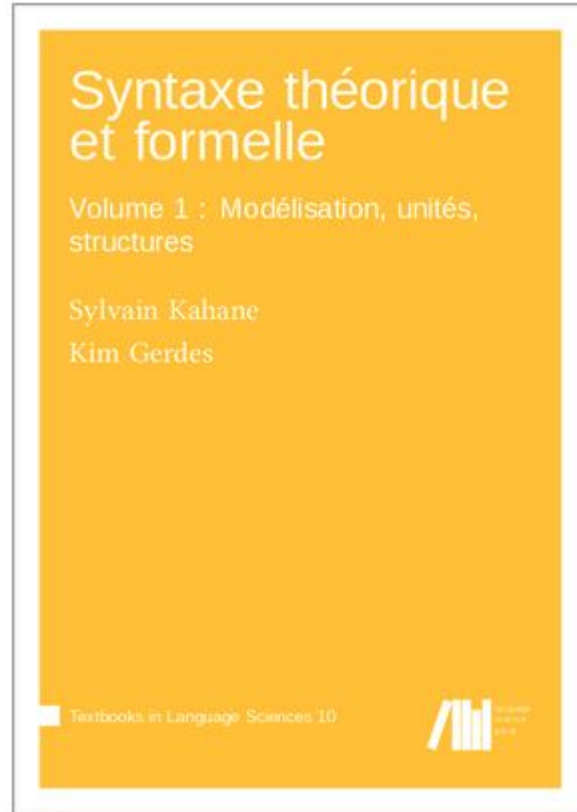
## Autre exemple d'une phrase non projective



Ignorez la ponctuation

En Grew-match : `global { is_not_projective }`

# Source



# DM1 noté

- Sur le drive (avec un corpus à utiliser)
- À rendre pour le 17 novembre
  - Par mail avec l'objet [M1 CAetP - DM1 / **NOM, Prénom**]
- Justifiez toutes vos réponses

→ **Questions ?**



# Schéma d'annotation SUD

## Specific SUD relations

SUD has 4 specific syntactic relations and a few extended relations:

- `subj`
- `udep`
- `comp`
  - `comp:aux`
  - `comp:cleft`
  - `comp:obj`
  - `comp:obl`
  - `comp:pred`
- `mod`

<https://surfacesyntacticud.github.io/guidelines/u/>

# Argument vs modifieur

- La plupart de sens ont une structure argumentale
  - Autrement dit, certains sens fonctionnent comme des prédicats qui exigent des arguments
  - Les arguments complètent le sens d'un prédicat
    - C'est pour ça qu'on les appelle compléments
  - Les modifieurs sont optionnels

Je bois trop d'eau

J'appelle mon fils tous les soirs

plat copieux

On parle de sous-catégorisation ou de valence d'une unité pour faire référence à l'ensemble de ses arguments

[X] **chez** Y  $\rightarrow$  Y = comp obligatoire

**question** de X à Y sur Z

# Annotation en ligne : Arborator

- <https://arboratorgrew.elizia.net>
- Projet = M1\_treebanks\_exos\_2023
- **intro\_sud-GSD**

