Analyse grammaticale

Béatrice Daille, Université de Nantes - LINA

2 avril 2013

- ⋆ **Étiquetage grammatical**
- ⋆ **Analyse syntaxique partielle**
- ⋆ **Analyse en constituants**
- ⋆ **Rappel linguistique - sous-catégorisation**
- ⋆ **Interface syntaxe-sémantique**

Étiquetage morpho-syntaxique

Parties du discours

Distribution des mots dans différentes classes grammaticales : parties du discours. Chaque mot de la langue a une **catégorie morpho-syntaxique** ou catégorie grammaticale.

- Classe ouverte : nom, adjectif, verbe, adverbe ;
- Classe fermée : déterminant, pronom, préposition (mots fonctionnels).

Étiquettes de parties du discours : 30-300. Faut-il distinguer :

- les noms communs des noms propres ;
- les pluriels des singuliers, le féminin du masculin ;
- les différents types de participe passé, les temps des verbes, etc.

Assigner à chaque mot sa partie du discours est une première étape de l'analyse syntaxique.

Ambiguïté

courant : verbe, nom, adjectif, préposition

la plupart du temps le contexte local suffit à désambiguïser

Étiquetage en parties du discours

Assigner à chaque mot sa catégorie grammaticale dans le contexte où il apparaît

Entrée : mot
Sortie : mot + catégorie grammaticale

Algorithme

1. assignation des étiquettes grammaticales du mot à l'aide d'un dictionnaire ou d'un analyseur morphologique
2. application de règles contextuelles pour choisir une étiquette

Règles contextuelles

- règles symboliques écrites manuellement
- suites d'étiquettes les plus probables dans un texte étiqueté - Modèle de Markov et algorithme de Viterbi
- règles symboliques apprises à partir d'un texte étiqueté - Apprentissage des transformations (Algorithme de Brill)

Modèle (1)

Objectif : trouver la meilleure suite d'étiquettes T pour une phrase S : $argmax_T p(T|S)$

Théorème de Bayes :
$p(T|S) = \frac{p(S|T)p(T)}{p(S)}$

On peut supprimer $p(S)$, si on est seulement intéressé par $argmax$:
$argmax_T p(T|S) = argmax_T p(S|T)p(T)$

$p(S|T) = p(w_1 \dots w_n | t_1 \dots t_n)$
Les mots sont indépendants les uns des autres étant donné une séquence d'étiquettes :
$p(S|T) = \prod_{i=1}^{n} p(w_i|t_1 \dots t_n)$
un mot dépend uniquement de son étiquette :
$p(S|T) = \prod_{i=1}^{n} p(w_i|t_i)$

Modèle (2)

Décomposition

$p(T) = p(t_1 \dots t_n)$
$p(T) = p(t_1)p(t_2|t_1)p(t_3|t_1t_2) \dots p(t_n|t_1 \dots t_{t-1})$
Une étiquette dépend uniquement des k étiquettes précédentes :
$p(T) = \prod_{i=1}^{n} p(t_i|t_{i-1} \dots t_{i-k})$
$k = 1$ modèle du 1^{er} ordre (bigramme)
$k = 2$ modèle du 2^{nd} ordre (trigramme)

$argmax_T \prod_{i=1}^{n} p(w_i|t_i)p(t_i|t_{i-1} \dots t_{i-k})$

estimation du modèle n-gramme à l'aide de l'estimation de vraisemblance maximale :
$p(w_i|t_i) = \frac{freq(w_i,t_i)}{freq(t_i)}$
$p(t_i|t_{i-1} \dots t_{i-k}) = \frac{freq(t_{i-k} \dots t_i)}{freq(t_{i-k} \dots t_{i-1})}$
avec :
$freq(w_i, t_i)$: nombre d'occurrences du mot w avec l'étiquette t
$freq(t_{l_1} \dots t_{l_m})$: nombre d'occurrences de la suite d'étiquettes $t_{l_1} \dots t_{l_m}$

Modèle de Markov caché (HMM)

Nous venons de définir une modèle de Markov caché (HMM)

HMM= (Q, O, T, E, $\mathbf{q}_1$)

Q un ensemble fini d'états q (ici les étiquettes)

O un alphabet fini de symboles en sortie (ici les mots)

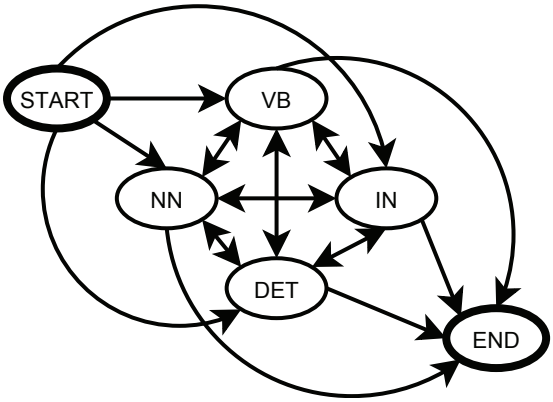
T un ensemble de probabilités de transition qui permettent de passer d'un état à l'autre (ici $p(t_n|t_{t-1})$))

E un ensemble de probabilités d'émission des symboles (ici $p(w_i|t_i)$)

$\mathbf{q}_1$ état initial (ici le début de la phrase)

HMM : représentation graphique

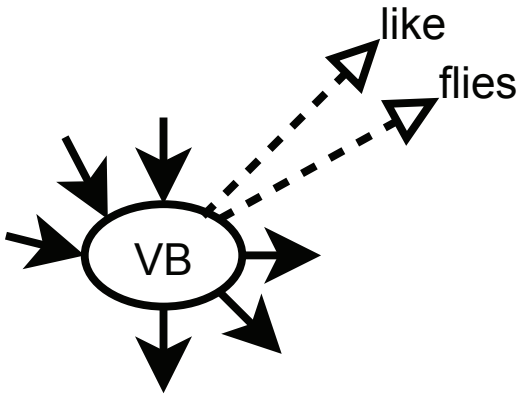
Lors de l'étiquetage d'une phrase en parties du discours, on chemine dans un graphe d'états



probabilités de transition d'un état à un autre : $p(t_n|t_{n-1})$

HMM : représentation graphique

Chaque état émet un mot :



probabilités d'émission des symboles : $p(w_i|t_i)$

Trouver la meilleur séquence d'étiquettes

⋆ Le modèle est défini. Comment l'utiliser ?

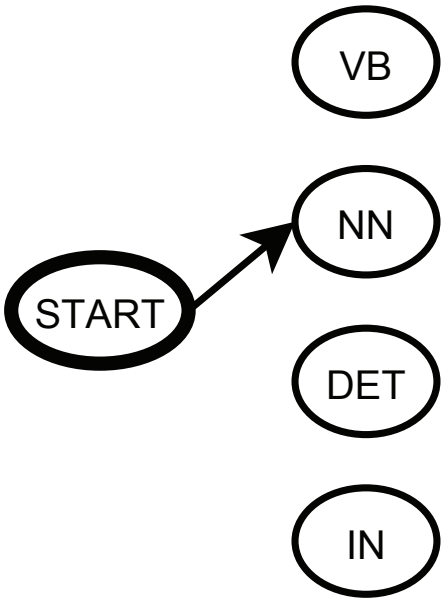
– ce qu'on a : une séquence de mots
  par exemple *Time flies like an arrow*
– ce que l'on veut obtenir : une séquence d'étiquettes
  Time/NN *flies*/VB *like*/P *an*/DET *arrow*/NN

⋆ si on considère une séquence d'étiquettes, il est facile de calculer sa probabilité :
$p(S|T)p(T) = \prod_i p(w_i|t_i)p(t_i|t_{i-1})$

⋆ Problème : si on a une moyenne de c étiquettes pour chacun de mots, il y a c^n séquences d'étiquettes possibles, cela peut être trop à évaluer.

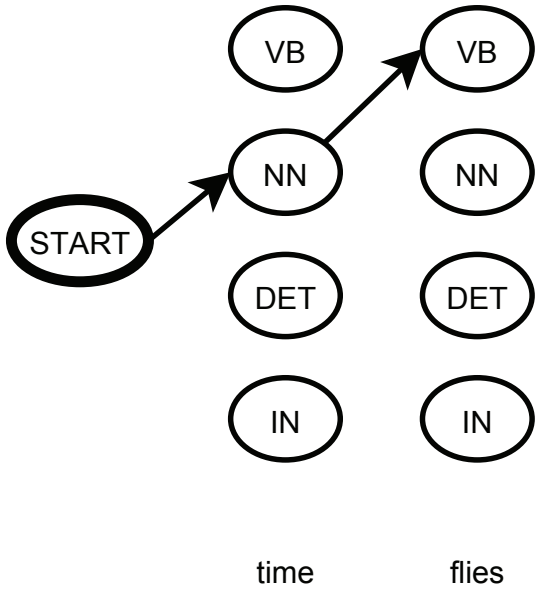
Cheminement dans les états (1)

Au début :



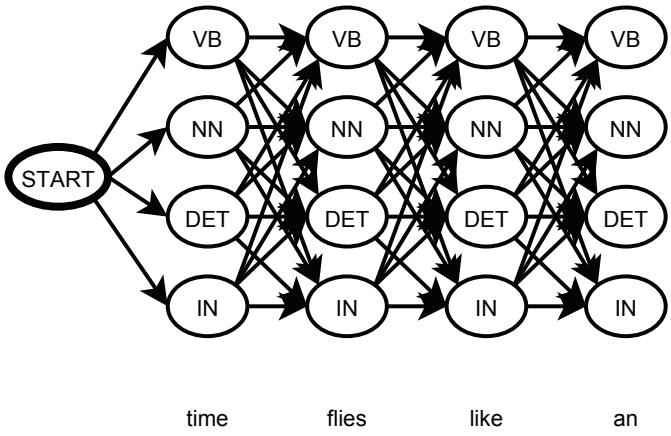
Cheminement dans les états (2)

Puis on se rend à l'état VB qui écrit la chaîne *flies*



Cheminement dans les états (3)

Bien entendu, il y a beaucoup de chemins possibles (treillis)



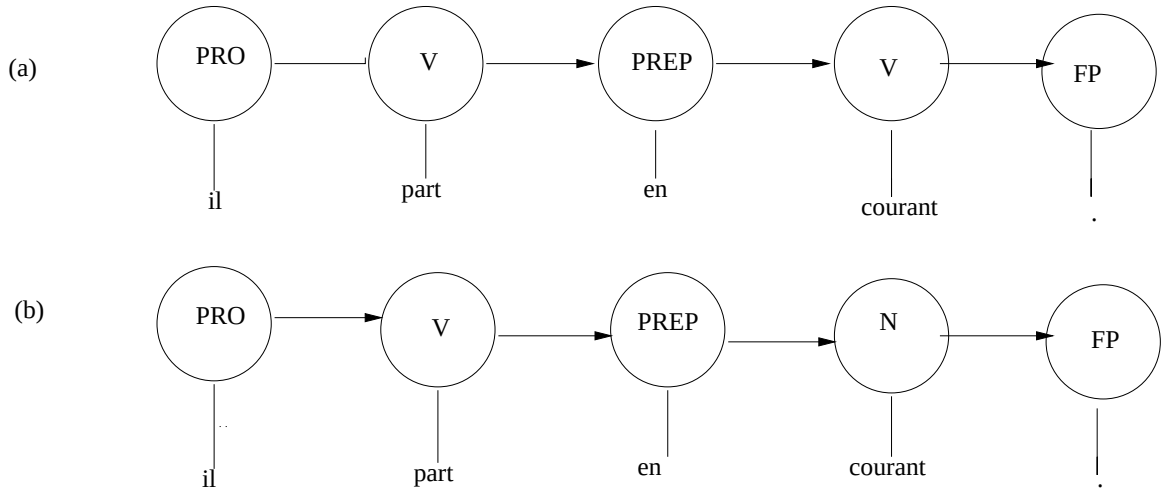
Algorithme de parcours d'un HMN

$X = (X_1, X_2, \ldots, X_{T+1})$ une séquence d'états

1. t := 1
2. Commencer à l'état q_i associé à une probabilité π_i (ie $X_1 = i$)
3. **tant que**
4. Passer de l'état q_i à l'état q_j avec la probabilité t_{ij} (ie $X_{t+1} = j$)
5. Écrire le symbole $o_t = k$ avec la probabilité b_{ij}
6. t := t+1
7. **fin**

Calcul de la probabilité d'un chemin

courant : V ou N
il part en courant.



$p(t_n|t_{t-1})$ (a) $p(V|PREP)$ = 0,83 (b) $p(N|PREP)$ = 0,00047
$p(w_i|t_i)$ (a) $p(courant|V)$ =0,00012 (b) $p(courant|N)$=0,00057
$p(t_n|t_{t-1})$ (a) $p(FP|V)$ = 0,0027 (b) $p(FP|N)$ = 0,012

$$p(V|PREP)p(courant|V)p(FP|V) = 0{,}00000027$$
$$p(N|PREP)p(courant|N)p(FP|N) = 0{,}000000000032$$

Algorithme de Viterbi

- Idée : Comme chaque transition ne dépend que de l'état courant, il est possible d'enregistrer à chaque état le meilleur chemin (variante de la distance d'édition)

- Est enregistré :
  - le coût moindre d'un état j à l'étape s $\delta_j(s)$
  - la trace arrière d'un état vers son meilleur prédécesseur $\psi_j(s)$
- En itérant parmi les états à chaque étape, calculer :
  - $\delta_j(s+1) = max_{1 \leq i \leq N} \delta_i(s) p(t_i|tj) p(w_s|t_i)$
  - $\psi_j(s+1) = argmax_{1 \leq i \leq N} \delta_i(s) p(t_i|tj) p(w_s|t_i)$
- Meilleur état final est $argmax_{1 \leq i \leq N} \delta_i(S+1)$, retour arrière à partir de cet état

Analyse syntagmatique

Syntaxe : étude des contraintes entre les catégories morphosyntaxiques pour former des phrases correctes

Les phrases ont une structure : elles sont constituées de **syntagmes ou constituants** :

Différents constituants :
- **groupe nominal** : *Traitement automatique du langage naturel*, *elle*, *le chien*
- **groupe adjectival** : *facile, très facile, facile à exécuter*
- **groupe verbal** : *manger beaucoup, a mangé, a mangé un gâteau*
- **groupe adverbial** : *très rapidement, de grande*

...

- tête du syntagme,
- compléments
- projection maximale

Grammaires formelles

G= ($\mathbf{V}_N$, $\mathbf{V}_T$, P, S)

$\mathbf{V}_N$ un vocabulaire auxiliaire,
$\mathbf{V}_T$ un vocabulaire terminal,
P un ensemble de règles de production,
S un symbole auxiliaire distingué (axiome)

Exemple de grammaire formelle

G= ($\mathbf{V}_N$, $\mathbf{V}_T$, P, S) avec :
$\mathbf{V}_N$ = {P, GN, GV, Det, N, V}
$\mathbf{V}_T$ = {Yann, aime, Laurie}
P :

1. GN $\rightarrow$ Det N
2. GV $\rightarrow$ V GN
3. P $\rightarrow$ GN GV

N. Chomsky *Syntactic Structures* (1957)
Hiérarchie des langages, différents types de grammaires formelles.

Ambiguïtés syntagmatiques

1. Attachement prépositionnel

Luc a rapporté un vase de Chine.

Luc a parlé de ce problème à Noël

2. Complétive/Relative

J'ai l'impression que vous ne mangez pas

Je comprends la forte impression qu'il vous a faite

3. Complément/Modifieur

Je mange la tarte

Je mange le matin

Analyseur partiel

Identifier certains syntagmes de la phrase sans réaliser une analyse complète de la phrase

Entrée : mot OU mot + catégorie grammaticale
Sortie : syntagmes non récursifs ou chunks

Chunk : patron prosodique constitué typiquement d'un mot simple et de mots fonctionnels
[Mais] [les déjeuners] [étaient hâtifs]
[à l'issue] [de la réunion] [de son cabinet]

Algorithme

1. identification des frontières et catégorisation du chunk à l'aide d'un dictionnaire de mots fonctionnels, d'une liste de séparateurs, ...
2. application de règles contextuelles qui dépendent de la catégorie du chunk.

Analyse syntaxique en constituants

Analyseurs pour des grammaires hors-contextes :
- descendants
- ascendants
- tabulaire ascendant - algorithme CKY
- tabulaire - algorithme d'Earley 1970

Analyseurs descendants
un même constituant est analysé plusieurs fois

Analyseurs ascendants
construction de tous les constituants possibles

Analyseurs tabulaires
analyse les constituants qu'une seule fois
utilisation d'une table (chart)

Analyseur tabulaire - CKY

Algorithme Cocke-Kasami-Younger (CYK)
Analyse ascendante
Entrée : Grammaire HC sous forme normale de Chomsky (FNC)
Stokage des constituants intermédiaires
Programmation dynamique

Forme normale de Chomsky (FNC)

Grammaire HC
G= ($\mathbf{V}_N$, $\mathbf{V}_T$, P, S) avec :

P :

1. A $\rightarrow$ a où a $\in$ V_T et A $\in$ V_N
2. A $\rightarrow$ B C où B, C $\in$ V_N

Pas de ϵ-production
Partie droite comporte soit un terminal, soit 2 non-terminaux

Table des sous-chaines bien formées

Table contient des analyses partielles

Exemple : *la bonne cuisine*
P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ; NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ; S $\rightarrow$ NP V

la	bonne	cuisine
Det_1 Pro_1	NP_4	
	N_2 Adj_2	
		N_3 V_3

Construction de la table (1)

Initialisation

Exemple : *la bonne cuisine*
P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0	1	2	3
0			
1			
2			

Construction de la table (2)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	[0,1]		
1		[1,2]	
2			[2,3]

Construction de la table (3)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	Det,Pro[0,1]		
1		[1,2]	
2			[2,3]

Construction de la table (4)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	Det,Pro[0,1]		
1		Adj,N [1,2]	
2			[2,3]

Construction de la table (5)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	**Det**,Pro**[0,1]**	**NP[0,2]**	
1		Adj,**N[1,2]**	
2			[2,3]

Construction de la table (6)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	Det, Pro[0,1]	NP[0,2]	
1		Adj, N[1,2]	
2			N, V[2,3]

Construction de la table (7)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	Det,Pro[0,1]	**NP[0,2]**	**S[0,3]**
1		Adj[1,2], N[1,2]	
2			N, **V[2,3]**

Construction de la table (8)

P : Det $\rightarrow$ la ; Pro $\rightarrow$ la ; Adj $\rightarrow$ bonne ; N $\rightarrow$ bonne ; N $\rightarrow$ cuisine ; V $\rightarrow$ cuisine ;
NP $\rightarrow$ Det N ; NP $\rightarrow$ Det NP1 ; NP1 $\rightarrow$ Adj N ;
S $\rightarrow$ NP V

0 0	la 1	bonne 2	cuisine 3
0	Det,Pro[0,1]	NP[0,2]	S[0,3]
1		**Adj[1,2]**, N[1,2]	**NP1[1,3]**
2			**N**, V[2,3]

Construction de la table (9)

P : Det → la ; Pro → la ; Adj → bonne ; N → bonne ; N → cuisine ; V → cuisine ;
NP → Det N ; NP → Det NP1 ; NP1 → Adj N ;
S → NP V

0 0	la 1	bonne 2	cuisine 3
0	**Det**,Pro**[0,1]**	NP[0,2]	**NP**, S[0,3]
1		Adj[1,2], N[1,2]	**NP1[1,3]**
2			N, V[2,3]

Algorithme CKY

FONCTION Analyseur-CKY(*mots*,*grammaire*)
RETURN *table*
FOR $j \leftarrow$ **DE** 1 **A** Nombre_de_mots(*mots*)
FAIRE
table[j-1,j] $\leftarrow$ {A|A $\rightarrow$ *mots*[j] $\in$ *grammaire*}
FOR $i \leftarrow$ **DE** j-2 **A** 0 **FAIRE**
FOR $k \leftarrow$ **DE** i+1 **A** j-1 **FAIRE**
table[i,j] $\leftarrow$ *table*[i,j] $\cup$
{A|A $\rightarrow$ *BC*[j] $\in$ *grammaire*,
B $\in$ *table*[i,k],
C $\in$ *table*[k,j]}
FIN FOR
FIN FOR
FIN FOR

Analyseur CKY : complexité

complexité en temps est $0(n^3)$ avec n nombre de mots de la phrase
la complexité augmente avec le nombre de règles de manière constante : $L = r^2$

Analyseur tabulaire - Earley

Utilisation de règles pointées

$X \rightarrow Y \ldots \bullet Z \ldots$

$P \rightarrow \bullet$ SN SV
$P \rightarrow$ SN $\bullet$ SV
$P \rightarrow$ SN SV $\bullet$

Arc de la table

<Début, Fin, Cat $\rightarrow$ trouvé $\bullet$ à_trouver>

arc passif tous les constituants ont été identifiés
constituants complets - hypothèse confirmée

arc actif possède une liste "à_trouver" non vide
constituants manquants - hypothèse à confirmer

Algorithme

1. **phase d'initialisation** qui introduit dans la table des arcs passifs
2. **règle fondamentale** qui permet de combiner entre eux arcs actifs et arcs passifs
3. **stratégie de contrôle** ascendante ou descendante
4. **stratégie d'exploration de l'espace de recherche** qui décide de l'ordre dans lequel sont traités les arcs de la table.

Initialisation

Ajout à la table de tous les arcs lexicaux correspondant aux unités lexicales de la phrase donnée en entrée

le chat danse.

Stratégie ascendante

Det $\rightarrow$ le $\bullet$

N $\rightarrow$ danse $\bullet$

N $\rightarrow$ chat $\bullet$

V $\rightarrow$ danse $\bullet$

Stratégie descendante

<1, 1, P $\rightarrow$ $\bullet$ CDroit>

Règle fondamentale

X et Y des séquences de n catégories

$A \rightarrow (X) \bullet B\ (Y)$
$B \rightarrow Z \bullet$

$A \rightarrow (X)\ B \bullet (Y)$

Règle de combinaison

$<i, j, A \rightarrow (X) \bullet B\ (Y) >$
$<j, k, B \rightarrow Z \bullet >$

$<i, k, A \rightarrow (X)\ B \bullet (Y) >$

Stratégie de contrôle

Ascendante
Présence d'un arc passif demande la sélection d'une règle syntagmatique dont le coin gauche est identique à la catégorie résultante de l'arc passif

$<i, j, C \rightarrow X \bullet >$ arc passif
$B \rightarrow C\ (Y)$

$<i, i, B \rightarrow \bullet\ C\ (Y) >$ arc actif

Descendante
Choix d'une règle se fait sur la base des arcs actifs : le premier constituant de la liste des constituants à trouver de l'arc actif doit être identique au coté gauche de la règle sélectionnée

$<i, j, C \rightarrow (X) \bullet Y>$ arc actif
$Y \rightarrow Z$

$<j, j, Y \rightarrow \bullet\ Z>$ arc actif

Stratégie d'exploration

Espace de recherche est constitué par l'ensemble des arcs

Stratégie décide de l'ordre dans lequel sont traités les arcs présents dans la table

Utilisation d'une structure intermédiaire dénommée AGENDA dans laquelle on stocke les arcs nouvellement crées
$\rightarrow$ assure que chaque arc est traité exactement une fois

PILE FILO $\rightarrow$ espace en profondeur
QUEUE FIFO $\rightarrow$ espace en largeur

Algorithme

1. Initialiser la table et l'agenda
2. Traiter agenda
3. Lorsque agenda est vide, examiner la table pour savoir si la phrase est acceptée ou rejetée. La phrase est acceptée ssi la table contient 1 ou plusieurs arcs s'étendant du début à la fin de la table et dont l'étiquette est de la forme : $P \rightarrow X \bullet$

Pour traiter agenda, répéter jusqu'à épuisement de l'agenda :

1. Retire le premier arc de l'agenda
2. Ajouter cet arc à la table (sauf si l'arc existe déjà)
3. Essayer de combiner cet arc avec des arcs présents dans la table par application de la règle fondamentale et ajouter arc résultants à l'agenda
4. Essayer de sélectionner de nouvelles règles et ajouter les arcs résultants à l'agenda.

Optimisation de la stratégie de sélection des règles

<i, j, C $\rightarrow$ (X) $\bullet$ Y>
Y $\rightarrow$ Z

<j, j, Y $\rightarrow \bullet$ Z>

Utilisation de la relation “est_un_coin_gauche_de”

<j, j, Y $\rightarrow \bullet$ Z> est ajouté à la table si un arc passif adjacent à droite <j, j+1, Cat $\rightarrow$ W> appartient à la table et est tel que Cat soit le coin gauche de Z

Rappel linguistique

Syntaxe : étude des contraintes entre les catégories morphosyntaxiques pour former des phrases correctes

Les phrases ont une structure : elles sont constituées de **syntagmes ou constituants** :
– tête du syntagme,
– compléments
– projection maximale

Entre les différents éléments lexicaux et syntagmes de la phrase, il existe des **contraintes** :
– accord
– auxiliaire
– sous-catégorisation

Sous-catégorisation (1)

sous-catégorisation = relation qu'entretient un prédicat et ses arguments, c'est à dire les constituants que le prédicat sélectionne.

(1) Max présente Luc à Marie

présenter sous-catégorise un sujet et un objet direct obligatoire, et un objet indirect (dit encore : second, datif, prépositionnel introduit par *à* ...) optionnel.

notion de complément obligatoire/optionnel,
notion de sélection sémantique

fonctions non sous-catégorisables/fonctions sous-catégorisables

fonctions sous-catégorisables :
– fonctions sémantiquement non-restreintes
– fonctions sémantiquement restreintes

Sous-catégorisation (2)

Non sous-catégorisables = modifieurs

modifieurs non phrastiques	une table *verte*, une table *en bois*, Max court *rapidement*, *très* souvent, *assez* rouge, *relativement* en retard, Max mange *avec une fourchette*,
modifieurs phrastiques à temps fini	la table *que j'ai achetée*, *Depuis que Max est là*, tout va mieux,
modifieurs phrastiques à l'infinitif	*Pour faire plus vite*, Max a traversé le champ

Sous-catégorisables = apparaissent dans la forme sémantique des prédicats :

Fonctions sémantiquement non restreintes : SUBJ, OBJ et l'objet indirect.
- l'agent : Max frappe Eva.
- la cause : La tempête détruit le bâteau.
- le patient : La branche casse.
- l'expériementateur : Max dort.
- le lieu : La Bretagne est pluvieuse.

Sous-catégorisation (3)

Fonctions sémantiquement restreintes :

- SCOMP : circonstance (Max veut *que tu partes*)
- VCOMP : circonstance (Max veut *partir*)

- OBL-BUT : Max va au Japon
- OBL-PRO : Max revient du Japon
- OBL-PL : Max habite au Japon
- OBL-MA : Max va bien
- OBL-AG : La souris a été mangée par le chat
- OBL-ME : Le mur mesure trois mètres
- ...

Structures de traits

La notion de base est celle de **trait**. Un trait est un couple attribut valeur.

trait (1) $\begin{bmatrix} \text{GEND} & \textbf{masc} \end{bmatrix}$

structure (2) $\begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{NB} & \textbf{sg} \\ \text{PER} & \textbf{3} \end{bmatrix}$

de trait

hiérarchie (3) $\begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{NB} & \textbf{sg} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix}$

chemin : localisation d'un couple attribut valeur dans la structure :
ACCORD/GEND = masc.

Subsomption

(3) $\begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{NB} & \textbf{sg} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix}$

(4) $\begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix}$

(3) est plus informatif que (4)
(4) est plus générique que (3).

(4) subsume (3) $\boxed{(4) \geq (3)}$

Relation d'ordre partiel : (4) et (5) ne sont pas en relation :

(4) $\begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix}$

(5) $\begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{NB} & \textbf{sg} \end{bmatrix} \end{bmatrix}$

Partage de valeurs

(7)
$$\begin{bmatrix} \text{CAT} & \textbf{s} \\ \text{ACCORD} & \boxed{1}\begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \\ \text{SUBJ} & \boxed{1} \end{bmatrix}$$

(8)
$$\begin{bmatrix} \text{CAT} & \textbf{s} \\ \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \\ \text{SUBJ} & \begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \end{bmatrix}$$

(7) : accord et subj partagent la même valeur
(8) : accord et subj ont chacun une copie de la valeur :

$$\begin{bmatrix} \text{NB} & \textbf{sg} \\ \text{GEND} & \textbf{fem} \end{bmatrix}$$

Partage de valeurs et subsomption

ST (7) et (8), qui subsume qui ?

(7) ne subsume pas (8)
(8) subsume (7), car

– soit $\boxed{1} = \begin{bmatrix} \text{NB} & \textbf{sg} \\ \text{GEND} & \textbf{fem} \end{bmatrix}$, ((7) = (8))

– soit $\boxed{1} \supset \begin{bmatrix} \text{NB} & \textbf{sg} \\ \text{GEND} & \textbf{fem} \end{bmatrix}$, ((7) est plus informatif que (8))

Partage de valeurs

(7)
$$\begin{bmatrix} \text{CAT} & \textbf{s} \\ \text{ACCORD} & \boxed{1}\begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \\ \text{SUBJ} & \boxed{1} \end{bmatrix}$$

(8)
$$\begin{bmatrix} \text{CAT} & \textbf{s} \\ \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \\ \text{SUBJ} & \begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \end{bmatrix}$$

(7) : accord et subj partagent la même valeur
(8) : accord et subj ont chacun une copie de la valeur :

$$\begin{bmatrix} \text{NB} & \textbf{sg} \\ \text{GEND} & \textbf{fem} \end{bmatrix}$$

Unification

Combinaison de deux structures de traits A et B, pour obtenir une nouvelle structure de trait C, plus spécifique que A et B : C est subsumé par A, et C est subsumé par B.

$$(4)\ \begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix}$$

$\wedge$

$$(5)\ \begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{NB} & \textbf{sg} \end{bmatrix} \end{bmatrix}$$

$$= (9)\ \begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{NB} & \textbf{sg} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix}$$

Contrainte : rendre l'unification unique

L'unification de A et B est la ST C la plus générale, telle que $C \leq A$ et $C \leq B$. On note $C = A \wedge B$.

Unification (2)

Processus récursif :

$$(10)\ \begin{bmatrix} \text{CAT} & \textbf{s} \\ \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{NB} & \textbf{sg} \end{bmatrix} \end{bmatrix}$$

$\wedge$

$$(11)\ \begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{CASE} & \textbf{nom} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix} =$$

$$\begin{bmatrix} \text{CAT} & \textbf{s} \\ \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{fem} \\ \text{PERS} & \textbf{3} \\ \text{CASE} & \textbf{nom} \\ \text{NB} & \textbf{sg} \end{bmatrix} \end{bmatrix}$$

Unification (3)

Cas d'échec :

- **Cas 1** Les valeurs atomiques sont différentes :

$$\begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{PER} & \textbf{3} \end{bmatrix} \end{bmatrix} \not\wedge \begin{bmatrix} \text{ACCORD} & \begin{bmatrix} \text{GEND} & \textbf{masc} \\ \text{PER} & \textbf{1} \end{bmatrix} \end{bmatrix}$$

- **Cas 2** Le partage de structures fait qu'il y a échec de valeurs atomiques.

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{ACCORD} & \boxed{\textbf{1}} \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{ACCORD} & \boxed{\textbf{1}} \end{bmatrix} \end{bmatrix}$$

$$\not\wedge$$

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{ACCORD} & [\ \text{NB} & \textbf{pl}\ ] \end{bmatrix} \\ \text{OBJ} & \begin{bmatrix} \text{ACCORD} & [\ \text{NB} & \textbf{sg}\ ] \end{bmatrix} \end{bmatrix}$$

Propriétés formelles :

- idempotence ($A \wedge A = A$)
- commutativité $A \wedge B = B \wedge A$
- associativité $(A \wedge B) \wedge C = A \wedge (B \wedge C)$
- Soit T (top) la ST moins informative que n'importe quelle autre ST, et $\perp$ (bottom) la ST virtuelle contenant plus d'information que n'importe quelle autre ST, alors $T \wedge A = A$, et $\perp \wedge A = \perp$.
- Interdéfinition $A \leq B$ si et seulement si $A \wedge B = A$

Algorithme d'unification

```
unif(f1, f2)
  si f1 [resp. f2] est une variable x alors
    si x est une variable libre alors
      # x n'a pas de valeur
      x := f2 [resp. f1]
    sinon     # (échec possible)
      unif(x, f2) [resp. unif(x, f1)]
    finsi
    renvoyer x
  finsi
  si f1 ou f2 est une valeur atomique alors
    si f1=f2 alors renvoyer f1
    sinon échec
  finsi
  # dernier cas : f1 et f2 sont nécessairement
  # des structures complexes
  res := [] # initialisation structure vide
  pour tout attribut T dans f1 ou f2 faire
    si {T:val1} appartient à f1 et
       {T:val2} appartient à f2 alors  # (échec possible)
      aux := unif(val1, val2)
      ajouter le trait {T:aux} à la structure res
    sinon
      # l'attribut T apparaît dans une seule structure
      soit {T:val} le trait appartenant à f1 ou f2
      ajouter {T:val} à res
    finsi
  finpour
  renvoyer res
```

Analyse avec contraintes d'unification

P	$\rightarrow$	NP	VP
	<NP ACCORD>	=	<VP ACCORD>
	<P TETE>	=	<VP TETE>

Adapation de l'analyseur tabulaire

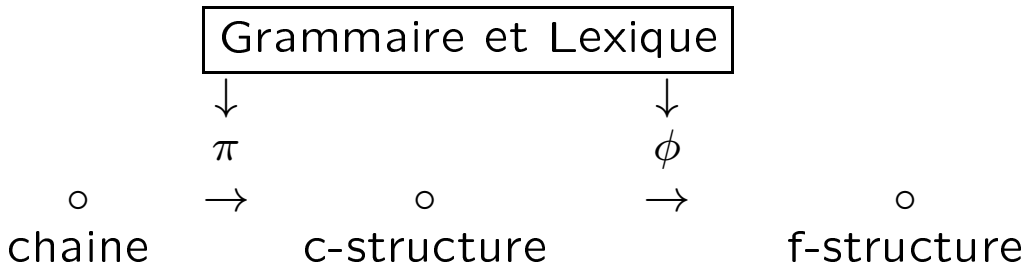
<Début, Fin, Cat $\rightarrow$ trouvé $\bullet$ à_trouver, ST>

Modification de la phase "combiner" dans "traiter agenda"

<i, j, A $\rightarrow$ (X) $\bullet$ B (Y), ST_A >
<j, k, B $\rightarrow$ Z $\bullet$, ST_B>

<i, k, A $\rightarrow$ (X) B $\bullet$ (Y) , UNIFICATION(ST_A, ST_B)>

LFG



Grammaire de type **hors contexte**

S →NP VP

Extension : les schémas

- codage de propriétés dans les entrées lexicales ("dormir" →nombre et nature syntaxico-sémantique de ses arguments, i.e. un sujet animé, et rien d'autre)
- transmission des informations entre deux nœuds quelconques.
- utilisation de ces informations pour construire et contraindre : (1) construire une structure fonctionnelle à partir de la c-structure, et (2) imposer des contraintes fonctionnelles de bonne formation sur les f-structures créées.

S → NP VP
(↑ SUBJ) = ↓ ↑ = ↓

- la partie gauche d'une règle (S) n'a pas de schéma associé.
- chaque symbole de la partie droite d'une règle (NP VP) est associé à 0, 1 ou plus schémas. Les schémas associés à un symbole sont situés sous le symbole.

Ensemble des schémas fonctionnels = **f-description**

LFG - Schémas fonctionnels

Différents types de schémas

Métavariables $\uparrow$ et $\downarrow$.
$\downarrow$: nœud associé au schéma dans la règle (f-structure associée au nœud.)
$\uparrow$: nœud père dans la règle.

$$
\begin{array}{lll}
X \rightarrow & Y & Z \\
 & & (\uparrow G) = val \\
 & & (\uparrow G) = \downarrow \\
 & & \uparrow = \downarrow \\
 & & (\uparrow G\ E) = (\downarrow F) \\
 & & (\uparrow(\downarrow G)\ ) = \downarrow
\end{array}
$$

alors :
- le schéma $(\uparrow G) = val$ signifie : " Le trait G de X a la valeur val",
- le schéma $(\uparrow G) = \downarrow$ signifie : " Le trait G de X a pour valeur l'ensemble de toutes les caractéristiques du nœud Z, i.e la f-structure associée au nœud Z".
- le schéma $\uparrow = \downarrow$ signifie : "X partage la f-structure associée à Z".
- le schéma $(\uparrow G\ E) = (\downarrow F)$ signifie : "Le trait E fait partie de la matrice valeur du trait G qui caractérise X. Le trait F caractérise Z. Et E et F partagent la même valeur."
- le schéma $(\uparrow(\downarrow G)\ ) = \downarrow$ signifie : "X est caractérisé par un trait dont l'identificateur, $(\downarrow G)$, est la valeur du trait G de Z. Ce trait a pour valeur la f-structure associée à Z"

Pas de relation directe entre deux frères (par l'intermédiaire du nœud père).

LFG - Exemple

Phrase à analyser : *Paul regarde l'écran*

Lexique minimal

Paul, N :	(↑ pred) = 'paul' (↑ nb) = sg
regarde, V :	(↑ pred) = 'regarder <(↑ subj), (↑ obj)>' (↑ subj nb) = sg
l, D :	(↑ det) = + (↑ nb) = sg
écran, N :	(↑ pred) = 'écran' (↑ nb) = sg

Remarque :
(↑ pred), est une *forme sémantique* :

"lemme-du-prédicat<(↑ g1), ..., (↑ gn)>",

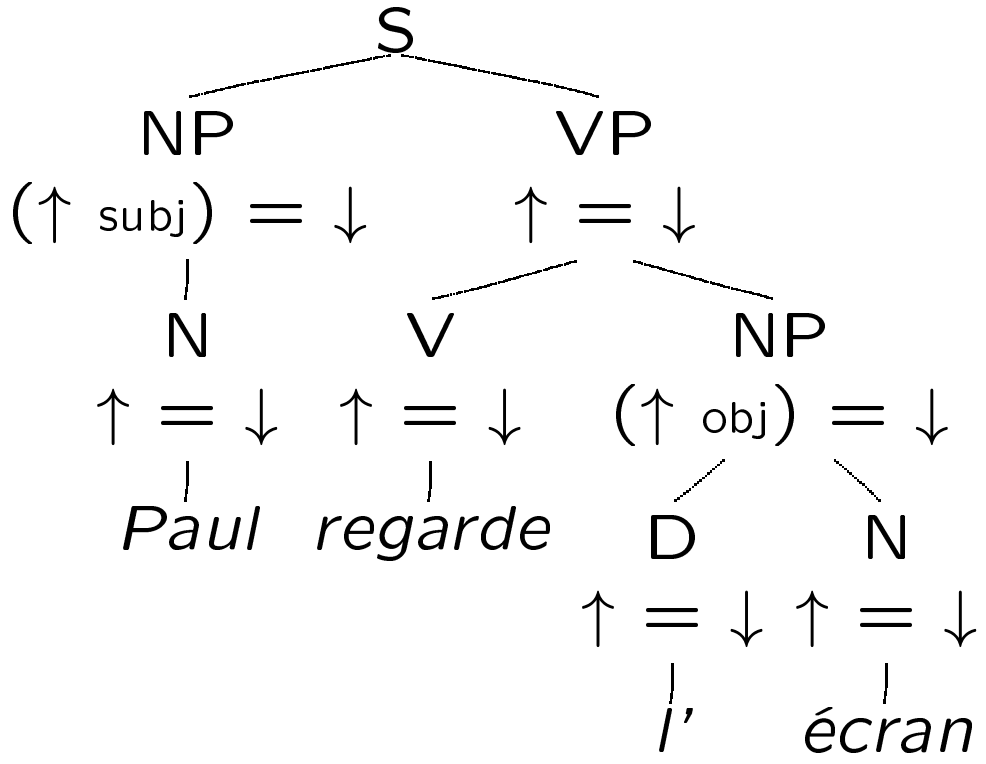
(↑ gi) = fonction grammaticale d'un argument du prédicat.

LFG - Exemple

Grammaire (G'1)

R1	$S \rightarrow$	NP $(\uparrow \text{SUBJ}) = \downarrow$	VP $\uparrow = \downarrow$
R2	$NP \rightarrow$	(D) $\uparrow = \downarrow$	N $\uparrow = \downarrow$
R3	$VP \rightarrow$	V $\uparrow = \downarrow$	NP $(\uparrow \text{OBJ}) = \downarrow$

C-structure



F-structure

$$\begin{bmatrix} \text{SUBJ} & \begin{bmatrix} \text{PRED} & \text{'Paul'} \\ \text{NB} & \textbf{sg} \end{bmatrix} \\ \text{PRED} & \text{'regarder } \langle \text{SUBJ, OBJ} \rangle \text{'} \\ \text{OBJ} & \begin{bmatrix} \text{PRED} & \text{'écran'} \\ \text{NB} & \textbf{sg} \\ \text{DET} & + \end{bmatrix} \end{bmatrix}$$