Grammaires hors contexte probabilistes (PCFG)

Carlos Ramisch

Sources

Ces supports réutilisent du matériel de :

- Diapos d'Alexis Nasr - Statistique Inférentielle 2015-2016
- C. Manning and H. Schütze, 1999, *Foundations of statistical NLP*, chapter 11
- D. Jurafsky and J. H. Martin, 2009, *Speech and Language Processing*, chapter 14

Programme

1. Langages et grammaires hors-contextes
2. Forme normale de Chomsky
3. Algorithme CYK
4. Grammaires hors contexte probabilistes
5. Probabilités *inside* et *outside*
6. Analyse syntaxique avec PCFG
7. Estimation des probabilités

Limites de l'hypothèse de Markov

Hypothèse de Markov en TAL

Un mot (événement) dans une phrase dépend uniquement des n mots précédents (historique récent)

Exemple : prédire le prochain mot avec un modèle à n-grammes

- *Jean **mange** des . . . pommes*
- *Jean **mange** souvent des **pommes***
- *Jean **mange** tous les matins des **pommes***
- *Jean **mange** tous les matins sauf les weekends des **pommes***
- Mais pour un modèle 3-grammes : *weekends des **? ? ?***

Limites de l'hypothèse de Markov

Hypothèse de Markov en TAL

Un mot (événement) dans une phrase dépend uniquement des n mots précédents (historique récent)

Exemple : prédire le prochain mot avec un modèle à n-grammes

- *Jean* ***mange*** *des . . .* ***pommes***
- *Jean* ***mange*** *souvent des* ***pommes***
- *Jean* ***mange*** *tous les matins des* ***pommes***
- *Jean* ***mange*** *tous les matins sauf les weekends des* ***pommes***
- Mais pour un modèle 3-grammes : *weekends des* ***? ? ?***

Limites de l'hypothèse de Markov

Hypothèse de Markov en TAL

Un mot (événement) dans une phrase dépend uniquement des n mots précédents (historique récent)

Exemple : prédire le prochain mot avec un modèle à n-grammes

- *Jean* ***mange*** *des . . .* ***pommes***
- *Jean* ***mange*** *souvent des* ***pommes***
- *Jean* ***mange*** *tous les matins des* ***pommes***
- *Jean* ***mange*** *tous les matins sauf les weekends des* ***pommes***
- Mais pour un modèle 3-grammes : *weekends des* ***? ? ?***

Limites de l'hypothèse de Markov

Hypothèse de Markov en TAL

Un mot (événement) dans une phrase dépend uniquement des n mots précédents (historique récent)

Exemple : prédire le prochain mot avec un modèle à n-grammes

- *Jean* ***mange*** *des . . .* ***pommes***
- *Jean* ***mange*** *souvent des* ***pommes***
- *Jean* ***mange*** *tous les matins des* ***pommes***
- *Jean* ***mange*** *tous les matins sauf les weekends des* ***pommes***
- Mais pour un modèle 3-grammes : *weekends des* ***? ? ?***

Limites de l'hypothèse de Markov

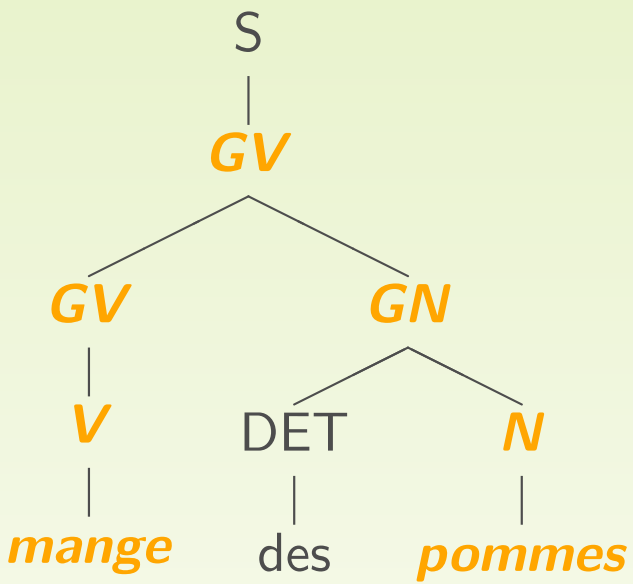
Hypothèse de Markov en TAL

Un mot (événement) dans une phrase dépend uniquement des n mots précédents (historique récent)

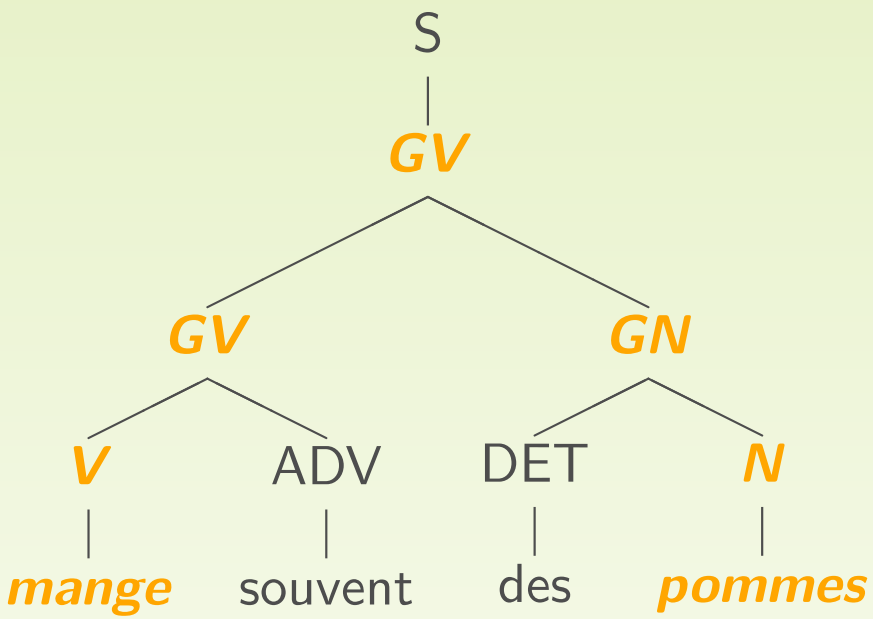
Exemple : prédire le prochain mot avec un modèle à n-grammes

- *Jean* ***mange*** *des . . .* ***pommes***
- *Jean* ***mange*** *souvent des* ***pommes***
- *Jean* ***mange*** *tous les matins des* ***pommes***
- *Jean* ***mange*** *tous les matins sauf les weekends des* ***pommes***
- Mais pour un modèle 3-grammes : *weekends des* ***? ? ?***

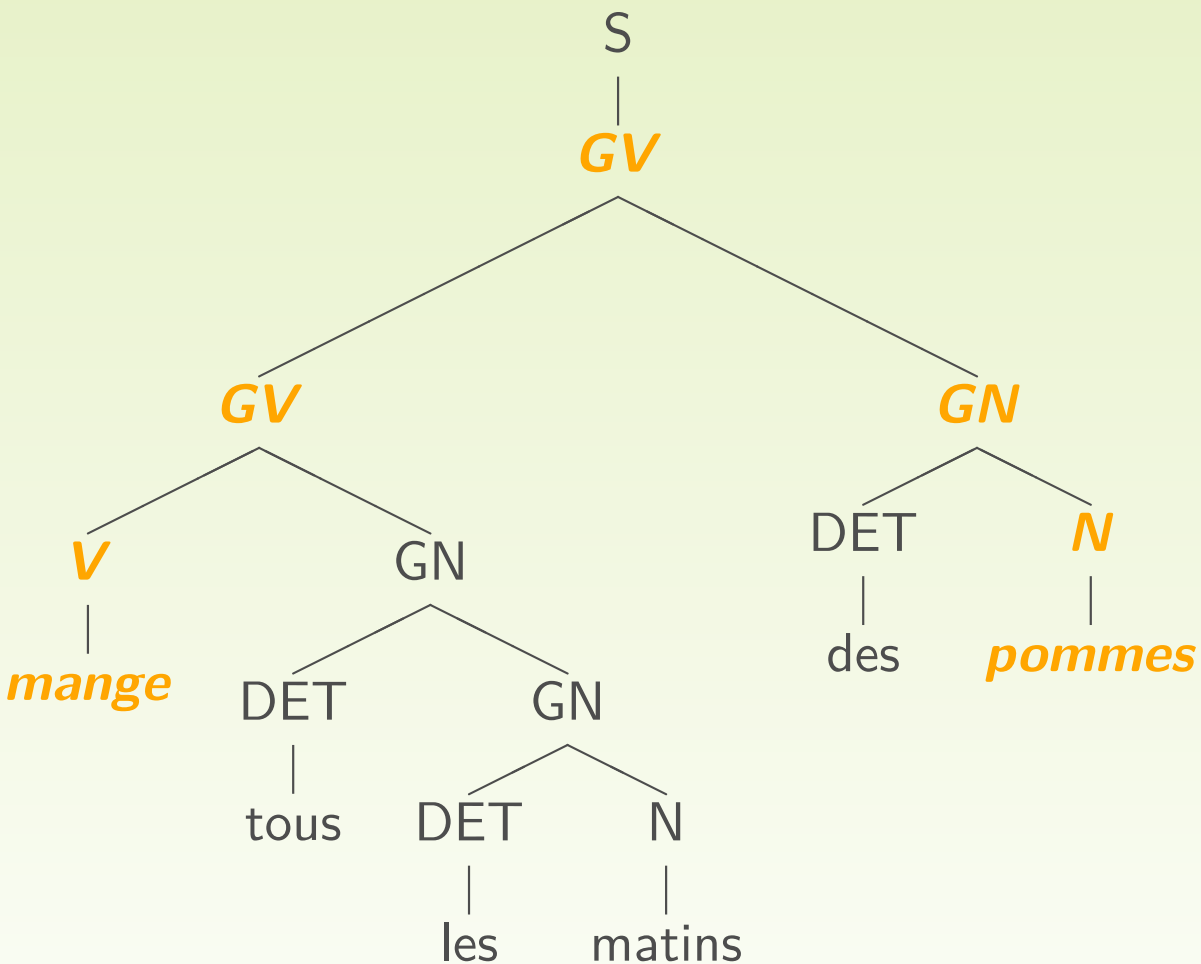
Analyse syntaxique de la phrase I



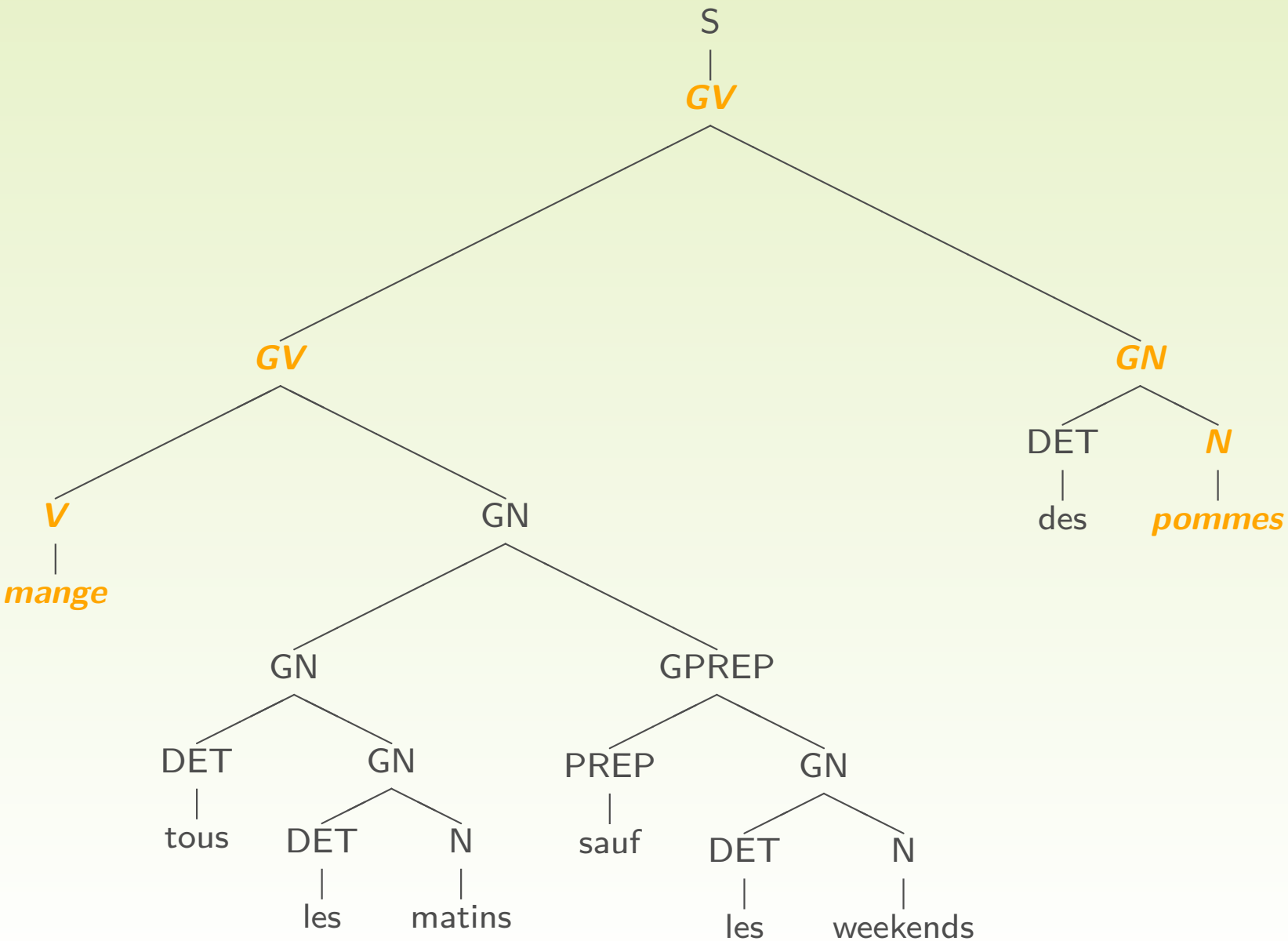
Analyse syntaxique de la phrase II



Analyse syntaxique de la phrase III



Analyse syntaxique de la phrase IV



Désambiguïsation syntaxique

- Parmi les analyses automatiques d'une phrase, la très grande majorité est aberrante.
- On aimerait pouvoir classer les analyses selon leur plausibilité.
- La grammaire probabiliste peut être utilisée comme modèle de désambiguïsation
- Elle permet de calculer la probabilité d'un arbre T étant donné une phrase S :

$$P(T|S, G)$$

- et de retrouver l'arbre le plus probable :

$$\hat{T} = \arg\max_{T} P(T|S, G)$$

Rappel : grammaires de réécriture

Une grammaire de réécriture est un 4-uplet $\langle N, \Sigma, P, S \rangle$ où :

- N est un ensemble de ***symboles non terminaux***, appelé l'***alphabet non-terminal***.
- Σ est un ensemble de ***symboles terminaux***, ***appelé l'alphabet terminal***, tel que N et Σ soient disjoints.
- P est un sous ensemble ***fini*** de :

$$(N \cup \Sigma)^* N (N \cup \Sigma)^* \times (N \cup \Sigma)^*$$

un élément (α, β) de P, que l'on note $\alpha \rightarrow \beta$ est appelé une ***règle de production*** ou ***règle de réécriture***.
α est appelé partie gauche de la règle
β est appelé partie droite de la règle

- S est un élément de N appelé l'***axiome*** de la grammaire.

Notation

Pour alléger les notations, on note :

$$\alpha \to \beta_1 \mid \beta_2 \mid \ldots \mid \beta_n$$

les n règles :

$$\alpha \to \beta_1 \ , \ \alpha \to \beta_2 \ , \ldots , \ \alpha \to \beta_n$$

Types de règles

- règles ***régulières*** à gauche :
  Une règle est régulière à gauche si et seulement si elle est de la forme $A \rightarrow xB$ ou $A \rightarrow x$ avec $A, B \in N$ et $x \in \Sigma^*$.
- règles ***hors-contexte*** :
  Une règle $A \rightarrow \alpha$ est un règle hors-contexte si et seulement si : $A \in N$ et $\alpha \in (N \cup \Sigma)^*$

Grammaires hors contexte (Context-Free Grammar - CFG)

Une grammaire G est ***hors contexte*** si toutes ses règles de production sont hors contexte.

Proto-mots d'une grammaire

Les ***proto-mots*** d'une grammaire $G = \langle N, \Sigma, P, S \rangle$ sont des mots construits sur l'alphabet $\Sigma \cup N$, on les définit récursivement de la façon suivante :

- S est un proto-mot de G
- si $\alpha\beta\gamma$ est un proto-mot de G et $\beta \to \delta \in P$ alors $\alpha\delta\gamma$ est un proto-mot de G.

Un proto-mot de G ne contenant aucun symbole non-terminal est appelé un mot engendré par G. Le ***langage engendré par*** G, noté $L(G)$ est l'ensemble des mots engendrés par G.

Dérivation

- L'opération qui consiste à générer un proto-mot $\alpha\delta\gamma$ à partir d'un proto-mot $\alpha\beta\gamma$ et d'une règle de production r de la forme $\beta \rightarrow \delta$ est appelée l'opération de ***dérivation***. Elle se note à l'aide d'une double flèche :

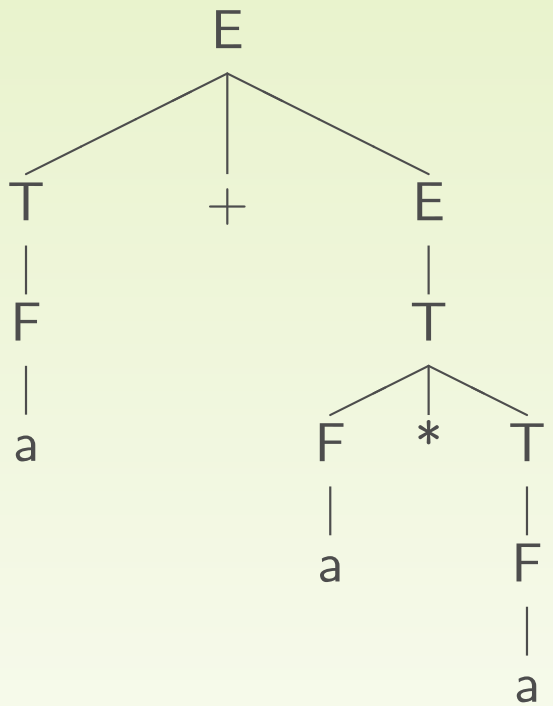
$$\alpha\beta\gamma \Rightarrow \alpha\delta\gamma$$

- On note $\alpha \overset{k}{\Rightarrow} \beta$ pour indiquer que β se dérive de α en k étapes.
- On définit aussi les deux notations $\overset{+}{\Rightarrow}$ et $\overset{*}{\Rightarrow}$ de la façon suivante :
  - $\alpha \overset{+}{\Rightarrow} \beta \equiv \alpha \overset{k}{\Rightarrow} \beta$ avec $k > 0$
  - $\alpha \overset{*}{\Rightarrow} \beta \equiv \alpha \overset{k}{\Rightarrow} \beta$ avec $k \geq 0$

Attention

Les symboles $\Rightarrow$ et $\rightarrow$ ne représentent pas la même chose.

Arbre de dérivation



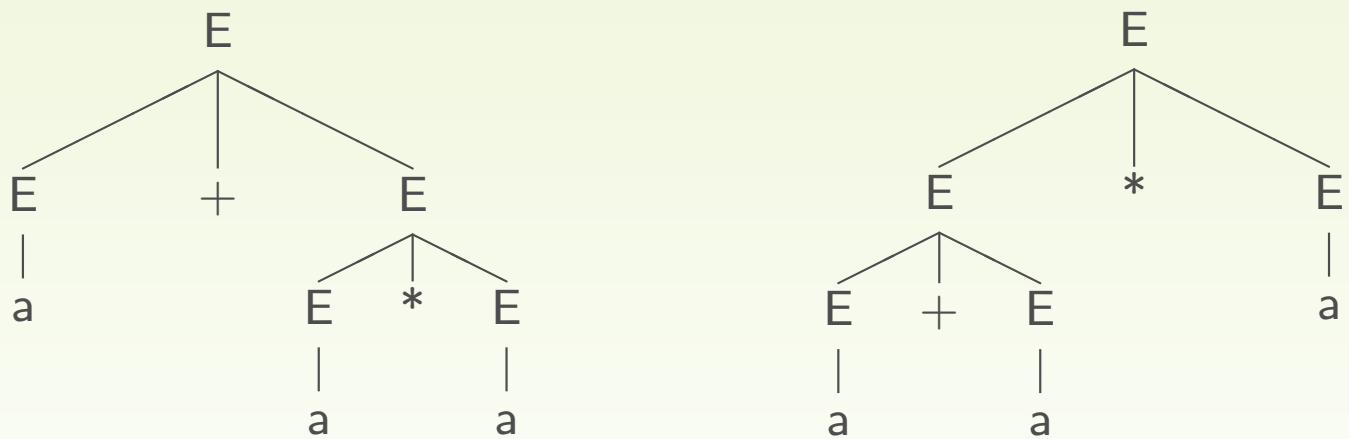
Un arbre de dérivation pour G ($G = \langle N, \Sigma, P, S \rangle$) est un arbre ordonné et étiqueté dont les étiquettes appartiennent à l'ensemble $N \cup \Sigma \cup \{\varepsilon\}$. Si un nœud de l'arbre est étiqueté par le non terminal A et ses fils sont étiquetés $X_1, X_2, ..., X_n$ alors la règle $A \rightarrow X_1, X_2, ..., X_n$ appartient à P.

Ambiguïté

Une grammaire G est ***ambiguë*** s'il existe au moins un mot m dans $L(G)$ auquel correspond plus d'un arbre de dérivation.

Exemple :

$$E \rightarrow E + E \mid E * E \mid a$$



Forme normale de Chomsky (CNF)

- Une CFG est en forme normale de Chomsky si toutes ses règles sont de la forme :

$$A \to BC \text{ ou } A \to a$$

avec $A, B \in N$ et $a \in \Sigma$. De plus, on autorise la règle $S \to \varepsilon$ si S est l'axiome de la grammaire et s'il n'apparaît jamais dans la partie droite d'une règle.

- Tout langage hors-contexte peut être généré par une grammaire hors-contexte en forme normale de Chomsky.

Conversion en CNF I

1. On crée un nouveau symbole S_0 et on ajoute la règle $S_0 \rightarrow S$. Cette modification garantit que l'axiome n'apparaît pas dans une partie droite de règle.
2. On élimine une règle de la forme $A \rightarrow \varepsilon \in P$, pour $A \neq S_0$ puis, pour toute occurrence de A dans une règle de P, on ajoute une nouvelle règle dans laquelle cette occurrence de A a été éliminée. La règle $X \rightarrow \alpha A \beta A \gamma$, par exemple, provoquera l'ajout des trois règles suivantes : $X \rightarrow \alpha \beta A \gamma$, $X \rightarrow \alpha A \beta \gamma$ et $X \rightarrow \alpha \beta \gamma$. Si $X \rightarrow A \in P$ alors on ajoute $X \rightarrow \varepsilon$ à moins que cette dernière n'ait déjà été éliminée. On recommence cette étape tant que P possède des règles-ε.

Conversion en CNF II

3. On élimine une règle de la forme $A \to B$. Pour toute règle de la forme $B \to \alpha$, on ajoute une règle $A \to \alpha$ à moins qu'il ne s'agisse d'une règle déjà éliminée. On recommence cette étape tant que P possède des règles de la forme $A \to B$.
4. Toute règle de la forme $A \to \alpha_1 \alpha_2 \dots \alpha_k$ avec $k \geq 3$ et $\alpha_i \in \Sigma \cup N$, est remplacée par les règles $A \to \alpha_1 A_1$, $A_1 \to \alpha_2 A_2$, $\dots$, $A_{k-2} \to \alpha_{k-1} \alpha_k$ où $A_1 \dots A_k$ sont de nouveaux non terminaux. Si $k \geq 2$, on remplace tout symbole terminal α_i des règles précédentes par un nouveau symbole non terminal U_i et on ajoute la règle $U_i \to \alpha_i$

Exercice

1. Transformez la grammaire ci-dessous en CNF :
   $S \rightarrow ASA|aB$
   $A \rightarrow B|S$
   $B \rightarrow b|\varepsilon$

Correction I

1. Création d'un nouvel axiome.
   $S_0 \to S$
   $S \to ASA|aB$
   $A \to B|S$
   $B \to b|\varepsilon$
2. Elimination des règles-ε.
   $S_0 \to S$
   $S \to ASA|aB|a|SA|AS|S$
   $A \to B|S$
   $B \to b$

Correction II

3. Elimination des règles $A \to B$.

Elimination de $S \to S$ et de $S_0 \to S$:
$S_0 \to ASA|aB|a|SA|AS$
$S \to ASA|aB|a|SA|AS$
$A \to B|S$
$B \to b$

Elimination de $A \to B$ et de $A \to S$:
$S_0 \to ASA|aB|a|SA|AS$
$S \to ASA|aB|a|SA|AS$
$A \to b|ASA|aB|a|SA|AS$
$B \to b$

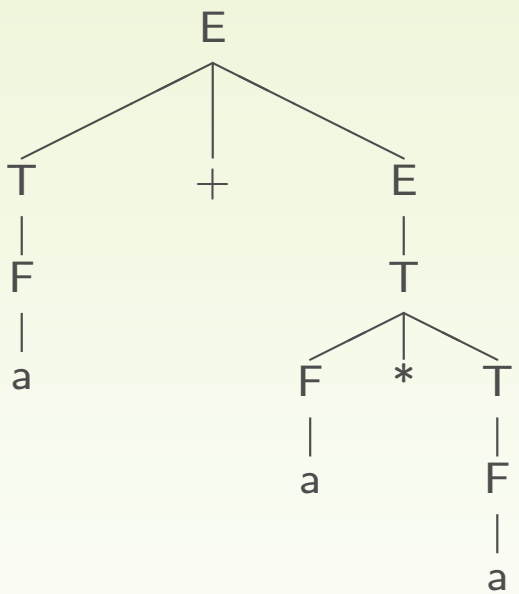
Correction III

4. Transformation des règles restantes :
   $S_0 \rightarrow AA_1|UB|a|SA|AS$
   $S \rightarrow AA_1|UB|a|SA|AS$
   $A \rightarrow b|AA_1|UB|a|SA|AS$
   $A_1 \rightarrow SA$
   $U \rightarrow a$
   $B \rightarrow b$

Analyse syntaxique

Etant donné $c \in \Sigma^*$ et $G = \langle \Sigma, N, P, A \rangle$, analyser c consiste à trouver pour c son (et éventuellement ses) arbre(s) de dérivation.

$$
\begin{array}{lcl}
c & = & a + a * a \\
 & & \\
G & = & \{ \\
\quad E & \rightarrow & T + E \mid T \\
\quad T & \rightarrow & F * T \mid F \\
\quad F & \rightarrow & (E) \mid a \\
\}
\end{array}
\quad \Longrightarrow
$$



Méthodes tabulaires

Programmation dynamique, les analyses partielles sont effectuées une seule fois et stockées dans une table.

Méthode	Espace	Temps
CYK	$\mathcal{O}(\|w\|^2)$	$\mathcal{O}(\|w\|^3)$
Earley	$\mathcal{O}(\|w\|^2)$	$\mathcal{O}(\|w\|^3)$

L'algorithme de Cocke-Younger-Kasami (CYK)

Entrée :

- une grammaire hors-contexte sous forme normale de Chomsky sans ϵ-transitions.
- une chaine $w = a_1 a_2 \ ... \ a_n \in \Sigma^+$

Sortie :

- Une table d'analyse T pour w telle que la case $t_{i,j}$ contient A si et seulement si $A \stackrel{+}{\Rightarrow} a_i a_{i+1} ... a_j$

Exemple

- Grammaire initiale

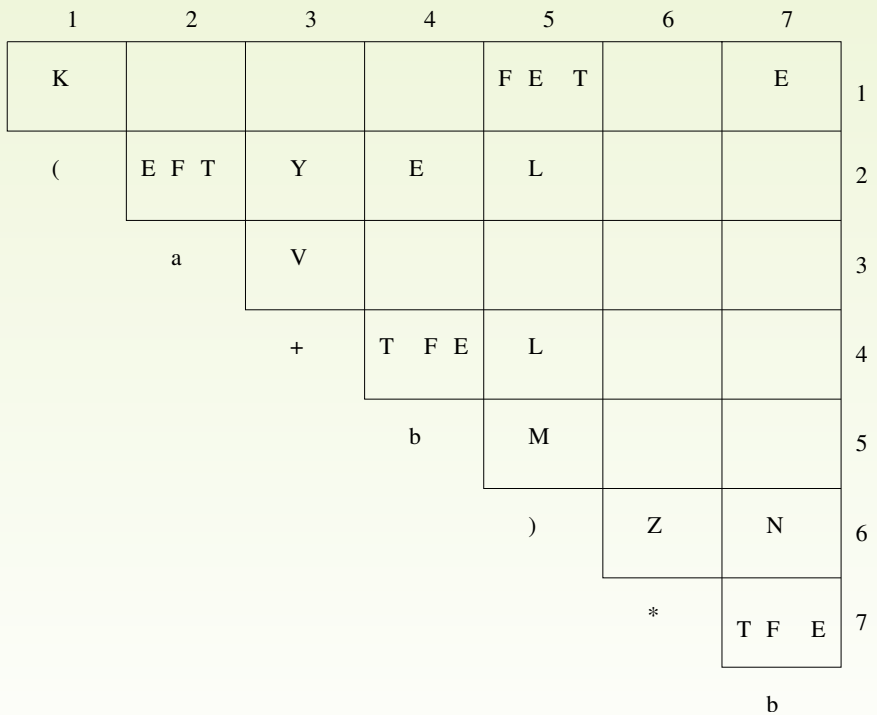
$$E \longrightarrow F + E | F$$
$$F \longrightarrow T * F | T$$
$$T \longrightarrow (E) | a | b$$

- Grammaire sous forme normale de Chomsky

$$E \rightarrow YE | TN | KL | a | b \quad N \rightarrow ZF \quad Z \rightarrow *$$
$$F \rightarrow TN | KL | a | b \quad L \rightarrow EM \quad K \rightarrow ($$
$$T \rightarrow KL | a | b \quad V \rightarrow + \quad M \rightarrow)$$
$$Y \rightarrow FV$$

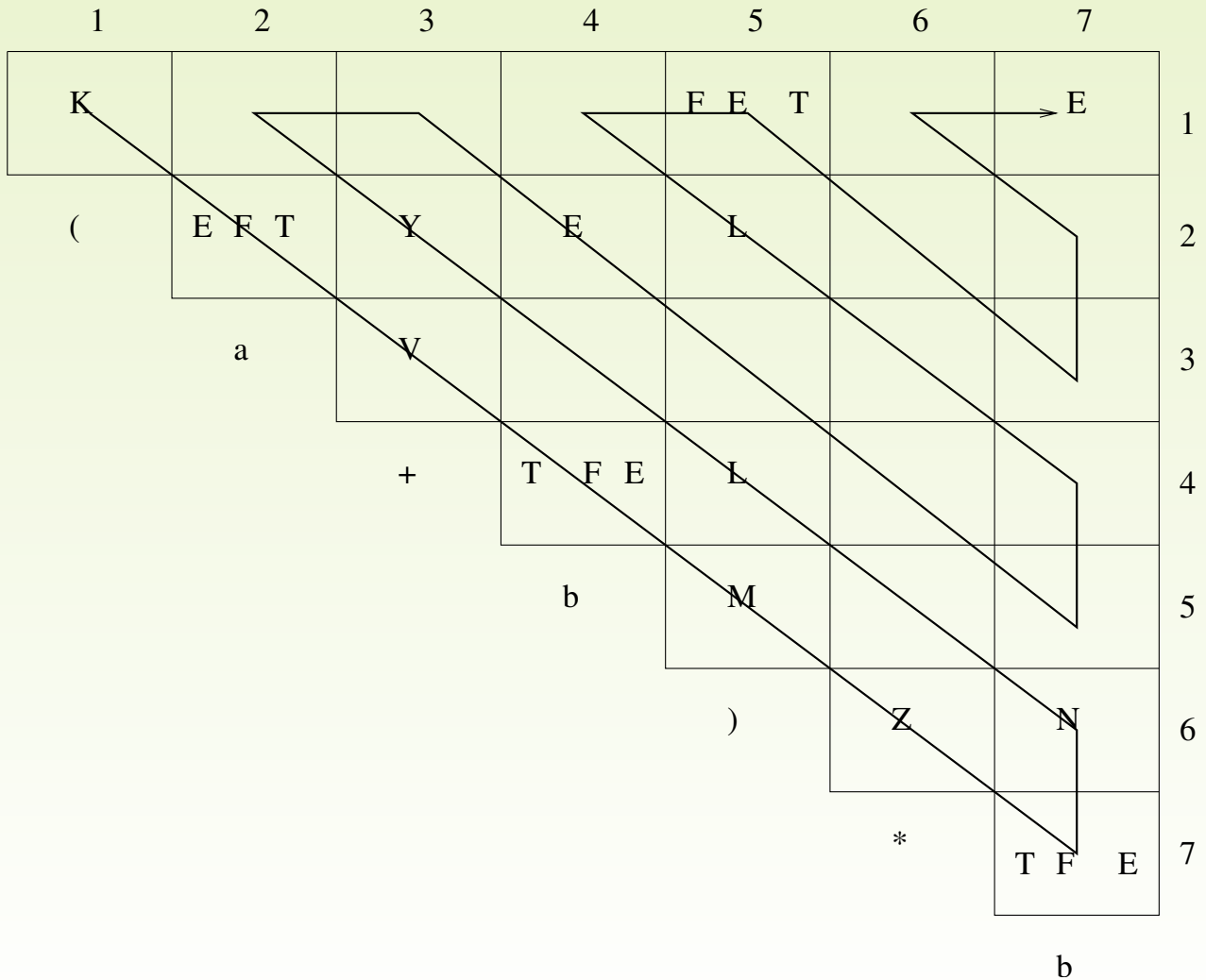
Analyse de la chaine $(a + b) * b$

$E \to YE|TN|KL|a|b \quad N \to ZF \quad Z \to *$
$F \to TN|KL|a|b \quad L \to EM \quad K \to ($
$T \to KL|a|b \quad V \to + \quad M \to)$
$Y \to FV$



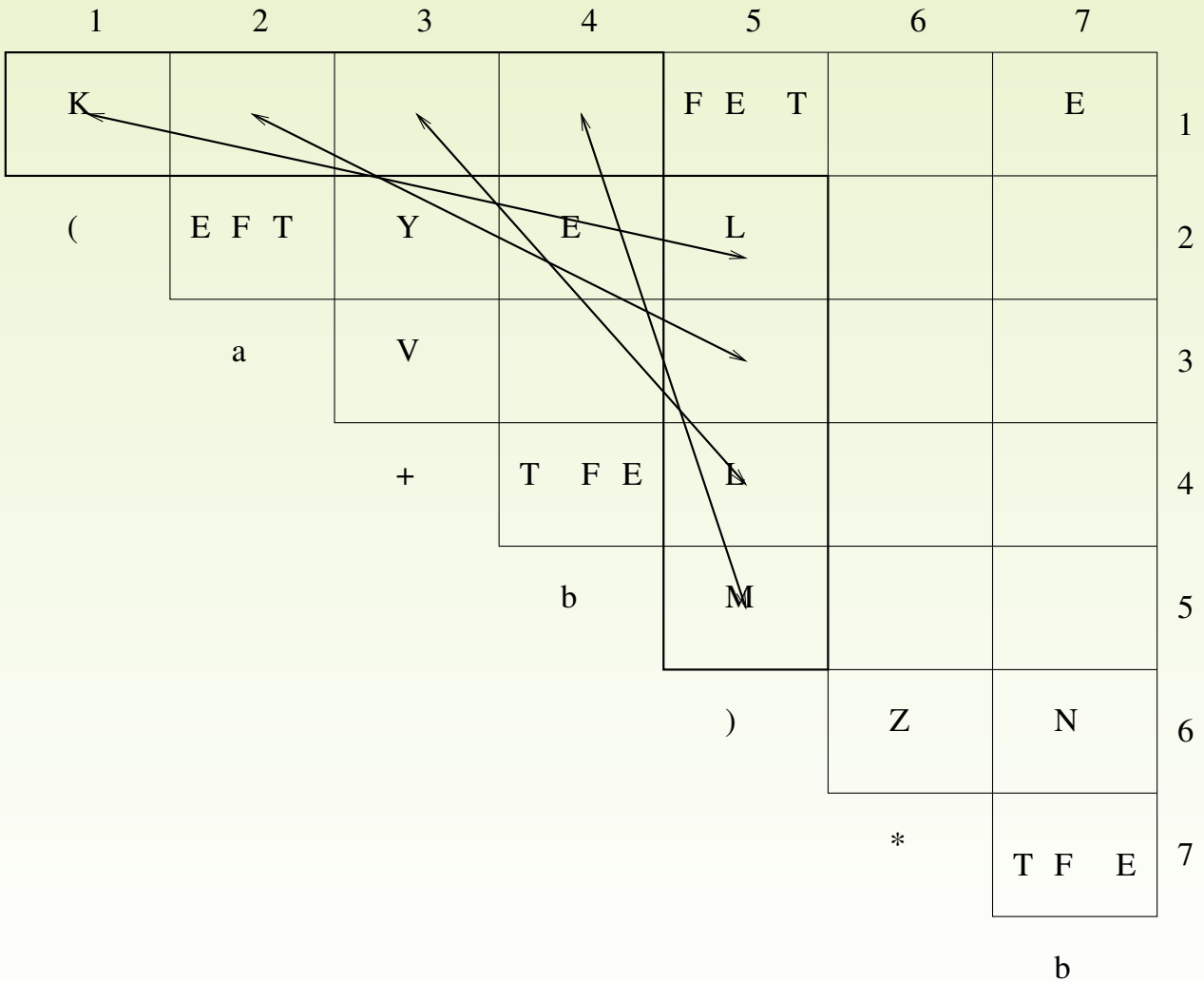
Analyse de la chaine $(a + b) * b$ I

Parcours du tableau :



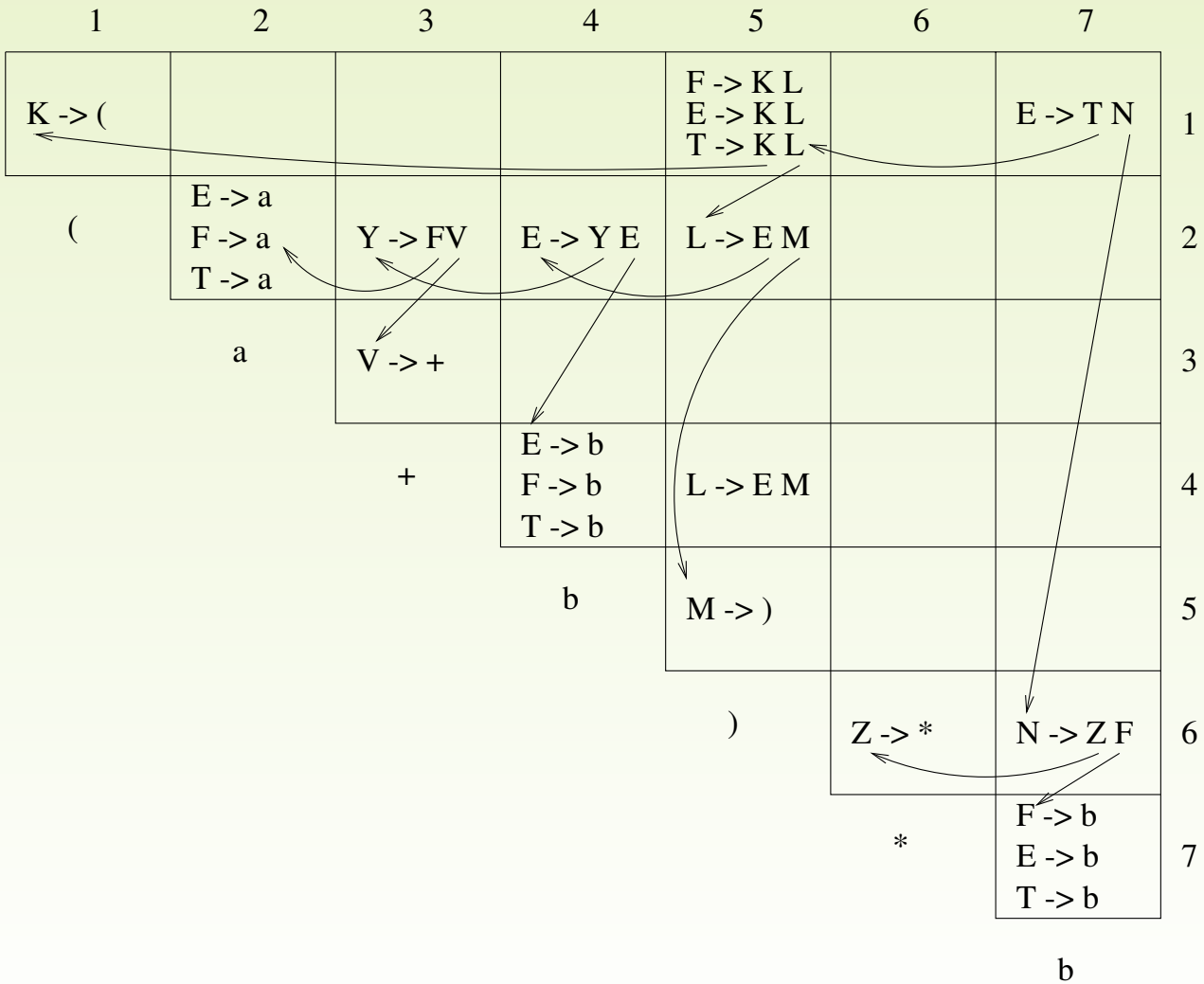
Analyse de la chaine $(a + b) * b$ II

Remplissage d'une case :



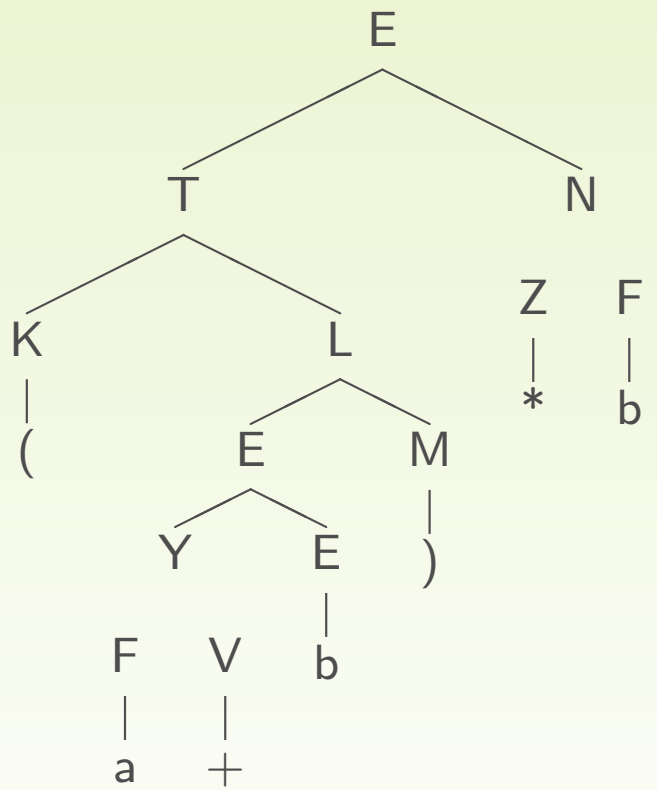
Analyse de la chaine $(a + b) * b$ III

Backtrack :



Analyse de la chaine $(a + b) * b$ IV

Résultat :



Algorithme CYK

pour $j = 1$ à n faire // INITIALISATION
  $t_{j,j} = \{A | A \rightarrow a_i\}$

pour $j = 2$ to n faire
  pour $i = j - 1$ downto 1 faire
    pour $k = i$ to $j - 1$ faire
      $t_{i,j} = t_{i,j} \cup \{A | A \rightarrow BC\}$
      avec $B \in t_{i,k}$ et $C \in t_{k+1,j}$

Grammaires hors-contexte probabiliste (PCFG)

Une grammaire hors-contexte probabiliste est composée de :

- Un alphabet non terminal $\mathcal{N} = \{N^1 \dots N^n\}$
- Un alphabet terminal $\mathcal{T} = \{t^1 \dots t^m\}$
- Un axiome N^1
- Un ensemble de règles $N^i \rightarrow \alpha$ avec $\alpha \in (\mathcal{N} \cup \mathcal{T})^*$
- Une distribution de probabilité associée à tout N^i :

$$\sum_j P(N^i \rightarrow \alpha^j) = 1$$

- On fera l'hypothèse que la grammaire est sous forme normale de Chomsky.

Probabilités

- La probabilité d'une règle est la probabilité de choisir cette règle pour réécrire le symbole de la partie gauche.

$$P(N^i \rightarrow \alpha^j) = P(N^i \rightarrow \alpha^j | N^i)$$

- Probabilité d'un arbre T :

$$P(T) = \prod_{n \in T} P(r(n))$$

où $n \in \mathcal{N}$ et $r(n)$ désigne la règle ayant permis de réécrire n.

- Probabilité d'une phrase S :

$$P(S) = \sum_{T \in \mathcal{T}(S)} P(T)$$

où $\mathcal{T}(S)$ est l'ensemble des analyses de S.

Trois problèmes à résoudre

1. Calcul de la probabilité d'une phrase S :

$$P(S|G) = \sum_{\forall T \in \mathcal{T}(S)} P(T, S|G)$$

2. Construction de l'arbre d'analyse de S le plus probable :

$$\hat{T} = \arg \max_{T \in \mathcal{T}(S)} P(T|S, G)$$

3. Estimation des probabilités de G à partir d'observations D :

$$\hat{G} = \arg \max_{G} P(D|G)$$

Parallèle avec les HMM

1. Calcul de la probabilité d'une séquence d'observations O :

$$P(O|\lambda) = \sum_{\forall Q} P(O, Q|\lambda)$$

2. Calcul du chemin le plus probable :

$$Q* = \arg\max_{Q} P(Q|O, \lambda)$$

3. Estimation des paramètres du HMM :

$$\hat{\lambda} = \arg\max_{\lambda} P(O|\lambda)$$

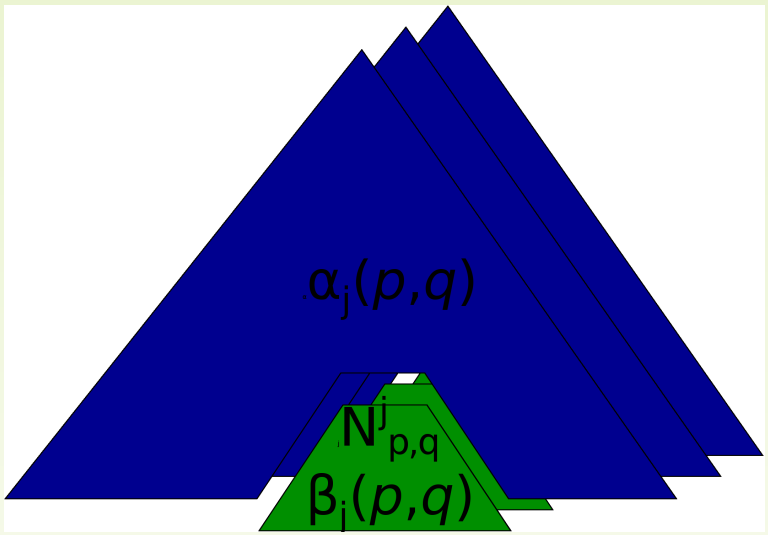
Différences

- Dans un HMM, une séquence d'états peut correspondre à plusieurs séquences d'observables alors qu'un arbre de dérivation correspond à une seule séquence d'observables (de terminaux).
- Étant donné un HMM et une séquence d'observables O, il est facile de déterminer tous les chemins ayant pu générer O. Dans le cas d'une grammaire probabiliste, il faut déterminer l'ensemble $\mathcal{T}(S)$: faire l'analyse syntaxique de S.

Notation

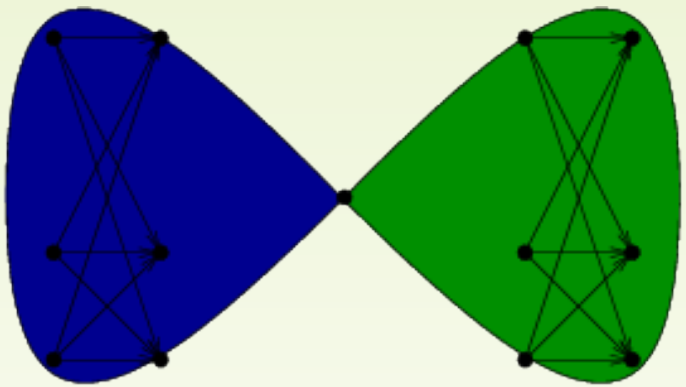
$m_1 \dots m_n$	phrase à analyser
$m_{p,q}$	segment $m_1 \dots m_n$ de la phrase
m^i	symbole de l'alphabet terminal
N^j	symbole de l'alphabet non terminal
$N^j_{p,q}$	le symbole N^j permet de dériver le segment $m_{p,q}$

Problème 1 - Probabilités extérieures et intérieures



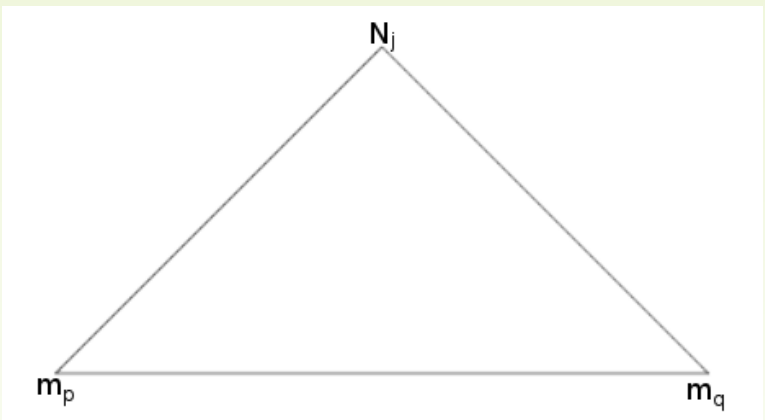
$$\beta_j(p,q) \stackrel{def}{=} P(m_{p,q}|N^j_{p,q}) \qquad \alpha_j(p,q) \stackrel{def}{=} P(m_{1,p-1}, N^j_{p,q}, m_{q+1,m})$$

Parallèle avec les HMM



Probabilité intérieure

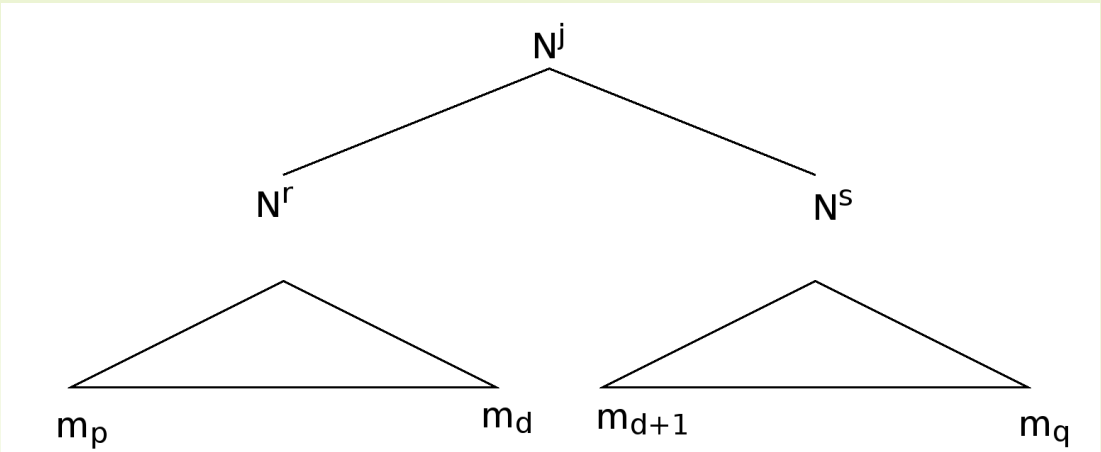
$$\beta_j(p,q) \stackrel{def}{=} P(m_{p,q}|N^j_{p,q})$$



Probabilité d'une phrase

$$
\begin{aligned}
P(m_{1,n}) &= P(N^1 \overset{*}{\Rightarrow} m_{1,n}) \\
&= P(m_{1,n} | N^1) \\
&= \beta_1(1, n)
\end{aligned}
$$

Calcul récursif des probabilités intérieures



- Calcul ascendant
- On calcule $\beta_j(p, q)$ à partir de $\beta_r(p, d)$ et $\beta_s(d + 1, q)$

Calcul récursif des probabilités intérieures

Relation de récurrence

$$
\begin{aligned}
\beta_j(p,q) &= P(m_{p,q}|N^j_{p,q}) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(m_{p,d}, N^r_{p,d}, m_{d+1,q}, N^s_{d+1,q}|N^j_{p,q}) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^r_{p,d}, N^s_{d+1,q}|N^j_{p,q}) P(m_{p,d}|N^r_{p,d}, N^s_{d+1,q}, N^j_{p,q}) \\
&\qquad\qquad P(m_{d+1,q}|N^r_{p,d}, N^s_{d+1,q}, N^j_{p,q}) \\
&= \sum_{r,s}\sum_{d=p}^{q-1} P(N^j \rightarrow N^r, N^s)\beta_r(p,d)\beta_s(d+1,q)
\end{aligned}
$$

Calcul récursif des probabilités intérieures

Cas terminal

$$\begin{aligned} \beta_j(k,k) &= P(m_k | N^j_{k,k}) \\ &= P(N^j \rightarrow m_k) \end{aligned}$$

Relation avec l'algorithme CYK

- $N^j_{p,q}$ correspond à la présence du symbole N^j dans la case $t_{p,q}$
- On peut calculer les $\beta(p, q)$ au fur et à mesure que l'on remplit la matrice d'analyse.

Calcul de $P(S)$ façon CYK

pour $q = 1$ à n faire // INITIALISATION
  pour $p = q$ à 1 faire
    si $(p == q)$
      $\beta_j(p, p) = P(N^j \rightarrow m_p)$
    sinon
      $\beta_j(p, q) = 0$
pour $q = 2$ à n faire
  pour $p = q - 1$ à 1 faire
    pour $d = p$ à $q - 1$ faire
      $\beta_j(p, q) = \beta_j(p, q) + P(N^j \rightarrow N^r, N^s)\beta_r(p, d)\beta_s(d + 1, q)$
      avec $N^r \in t_{p,d}$ et $N^s \in t_{d+1,q}$
$P(S) = \beta_1(1, n)$

Problème 2 - Trouver $\hat{T}$ le plus probable

$\delta_i(p, q) =$ la probabilité du sous-arbre $N^j_{p,q}$ le plus probable.

1. Initialisation

$$\delta_i(p, p) = P(N^i \to m_p)$$

2. Récurrence

$$\delta_i(p, q) = \max_{1 \leq j,k \leq n,\ p \leq d < q} P(N^i \to N^j N^k)\delta_j(p, d)\delta_k(d + 1, q)$$

3. Fin

$$P(\hat{T}) = \delta_1(1, n)$$

Calcul de $P(\hat{T})$

pour $q = 1$ à n faire { INITIALISATION }
  pour $p = q$ à 1 faire
    si $(p == q)$
      $\delta_j(p, p) = P(N^j \rightarrow m_p)$
    sinon
      $\delta_j(p, q) = 0$
pour $q = 2$ à n faire
  pour $p = q - 1$ à 1 faire
    pour $d = p$ à $q - 1$ faire
      $\delta_i(p, q) = \max(\delta_i(p, q), P(N^i \rightarrow N^j N^k)\delta_j(p, d)\delta_k(d + 1, q))$
      avec $N^r \in t_{p,d}$ et $N^s \in t_{d+1,q}$
$P(\hat{T}) = \delta_1(1, n)$

Construction de $\hat{T}$

$\psi_i(p,q) = \langle j,k,r \rangle$ où j,k,r désignent l'application de la règle ayant réalisé le maximum $\delta_i(p,q)$

$$\psi_i(p,q) = \arg\max_{j,k,d} P(N^i \rightarrow N^j N^k)\delta_j(p,d)\delta_k(d+1,q)$$

- racine($\hat{T}$) $= N^1_{1,n}$
- si $\psi_i(p,q) = \langle j,k,r \rangle$ alors
  - fils gauche($N^i_{p,q}$) $= N^j_{p,r}$
  - fils droit($N^i_{p,q}$) $= N^k_{r+1,q}$

Exercice I

1. Étant donné la grammaire ci-dessous, trouvez l'analyse la plus probable $\hat{T}$ pour la phrase $S =$*the flight includes a meal*

S	$\rightarrow$	$NP \;\; VP$	0.8
NP	$\rightarrow$	$DET \;\; N$	0.3
VP	$\rightarrow$	$V \;\; NP$	0.2
V	$\rightarrow$	*includes*	0.05
DET	$\rightarrow$	*the*	0.4
DET	$\rightarrow$	*a*	0.4
N	$\rightarrow$	*meal*	0.01
N	$\rightarrow$	*flight*	0.02

Exercice II

	the 1	flight 2	includes 3	a 4	meal 5
1	DET : 0.4	NP : .3 * .4 * .02 = 24e-4			S : .8 * 12e-6 * 24e-4 = 23e-9
2		N : .002			
3			V : 0.05		VP : .2 * .05 * 12e-4 = 12e-6
4				DET : 0.4	NP : .3 * .4 * .01 = 12e-4
5					N : 0.01

Problème 3 - Estimation des probabilités de la grammaire

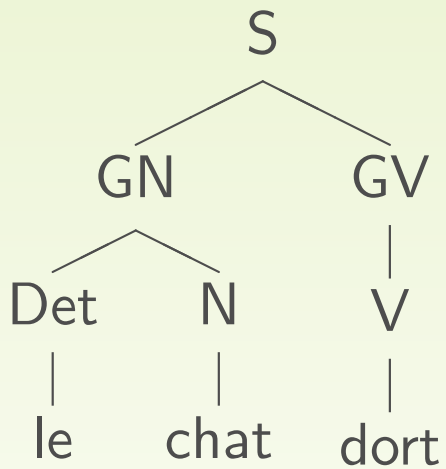
Deux cas :

- Données complètes : on dispose d'un ensemble de phrases et de leur analyse syntaxique (banque d'arbres).
- Données incomplètes : on ne dispose que d'un ensemble de phrases.

Exemples de banques d'arbres

corpus	Penn Treebank	corpus Paris 7
mots	1 000 000	400 000
phrases	45 000	15 000
cat. syntag.	26	13
parties de discours	36	14
règles	9 657	

Construction de la grammaire



$$
\begin{array}{lcl}
S & \rightarrow & GN\ GV \\
GN & \rightarrow & Det\ N \\
GV & \rightarrow & V \\
Det & \rightarrow & le \\
N & \rightarrow & chat \\
V & \rightarrow & dort
\end{array}
$$

Estimation des probabilités des règles

- On compte le nombre d'occurrences des symboles non terminaux A dans la banque d'arbres : $C(A)$
- On compte le nombre d'occurrences des règles $A \rightarrow \alpha$ dans la banque d'arbres : $C(A \rightarrow \alpha)$
- On estime $P(A \rightarrow \alpha)$ par maximum de vraisemblance :

$$P(A \rightarrow \alpha) = \frac{C(A \rightarrow \alpha)}{C(A)}$$

Estimation à partir de données incomplètes

- On fixe la partie algébrique de la grammaire (alphabets, règles)
- On recherche les distributions de probabilités de la grammaire qui maximise la probabilité des données d'apprentissage (des phrases dont on dispose)
- On ne sait calculer ces distributions directement, on fait appel à un algorithme itératif du type Expectation Maximization appelé algorithme intérieur-extérieur (inside-outside).

Principe de l'algorithme intérieur-extérieur

- On fixe des distributions de probabilités initiales
- On calcule la probabilité des différents symboles et règles étant donné une phrase S
- On estime le nombre d'occurrences des différents symboles et règles lors des dérivations de S
- On calcule de nouvelles distributions de probabilités
- On réitère tant que les nouvelles probabilités augmentent $P(S)$

Algorithme intérieur-extérieur

- Les formules de ré-estimation des poids dépendent des probabilités intérieur et extérieur
- Bien que simples, ces probabilités ont des formulations très longues à dériver (surtout α)
- Les détails de cet algorithme ne seront pas vus en cours
- Lecture recommandée : Manning & Schütze (1999), chapitre 11 - pages 381 à 405

Problèmes de l'algorithme intérieur-extérieur

- Efficacité : pour chaque phrase du corpus d'apprentissage et chaque itération, la complexité est $O(n^3 V^3)$ où n est la longueur de la phrase et V le nombre de non terminaux de la grammaire.
- Maximum local : l'algorithme est très sensible aux distributions de probabilités initiales. Des distributions différentes aboutissent à des maxima différents.
- Choix du nombre de non terminaux : on ne sait pas combien de non terminaux choisir, les expériences ont montré qu'un nombre important de non terminaux améliore les résultats, ce qui augmente le problème d'efficacité.

Limites des PCFG (1)

- Indépendance lexicale
  - La réecriture d'un symbole pré-terminal X ne dépend pas du contexte d'occurrence de X
  - mise en défaut : la préposition introduisant un complément d'un verbe dépend de la nature lexicale du verbe
    Exemple : *Jean* **pense à** *Marie*
  - cette dépendance n'est pas modélisée :
    $GV \rightarrow V\ GP$
    $V \rightarrow pense$
    $GP \rightarrow P\ GN$
    $P \rightarrow à$

Limites des PCFG (2)

- Indépendance structurale
  - le choix d'une règle pour la réécriture d'un symbole X est indépendant du contexte d'occurrence de X
  - mise en défaut : Dans le corpus switchboard, 91% des sujets sont des pronoms alors que seule 34% des objets le sont (Francis et al 99)
  - Or il n'y a qu'une règle de la forme $GN \rightarrow Pro$
  - et une seule probabilité lui correspondant