4 – Lambda-calcul et composition du sens

1. La représentation du sens en logique du premier ordre
2. Le principe de compositionnalité
3. Le lambda-calcul
4. L’analyse sémantique

4.1 – La représentation du sens en logique du premier ordre

- On se propose ici de représenter le **sens littéral** des énoncés langagiers, celui qui est indépendant du contexte dans lequel les énoncés sont produits.

- On peut utiliser différents cadres formels pour représenter le sens des énoncés : des **graphes ou réseaux sémantiques**, diverses **logiques**.

- La **logique du premier ordre** permet d'exprimer le sens d'un énoncé en termes de **conditions de vérité** à l'aide de la **théorie des modèles**.

- Elle permet d'effectuer des **inférences** sur des énoncés langagiers. Enfin, son **pouvoir d'expression** est suffisamment grand pour représenter les énoncés les plus courants.

4.1 – La représentation du sens en logique du premier ordre

- La **logique du premier ordre** est une extension de la **logique des propositions**.

- La **logique des propositions** opèrent sur des propositions, c'est-à-dire des objets qui ont la propriété d'être vrais ou faux. A l'aide de **connecteurs logiques** ($\neg, \wedge, \vee, \Rightarrow$) et à partir des propositions, elle construit des **formules logiques** dont les **valeurs de vérité** sont calculées à partir de celles des propositions élémentaires à l'aide de **tables de vérité**.

- La **logique des prédicats** remplace les propositions par des **prédicats**. Un prédicat requiert un nombre fixe d'**arguments** qui sont des termes pour prendre une valeur de vérité. Les termes peuvent contenir des **variables** et on ajoute les deux **quantificateurs**, existentiel et universel, comme moyen de former des formules logiques complexes.

4.1 – La représentation du sens en logique du premier ordre

- Exemple : base de données généalogique

Jean Muller a comme père Pierre Muller.	nom(i1,muller) $\wedge$ prenom(i1, jean) $\wedge$ nom(i2,muller) $\wedge$ prenom(i2, pierre) $\wedge$ pere (i1, i2)
Jean Muller n'est pas une femme.	nom(i1,muller) $\wedge$ prenom(i1,jean) $\wedge$ $\neg$ femme(i1)
Tout père est un homme	$\forall$ x $\forall$ y (pere(x,y) $\Rightarrow$ homme(x))
Tout homme a un père et une mère.	$\forall$ x (homme (x) $\Rightarrow$ $\exists$ y pere (y, x)) $\wedge$ $\exists$ z mere (z, x))
Tout parent d'un individu quelconque est un homme ou une femme	$\forall$ x ($\exists$ y parent (x,y) $\Rightarrow$ homme(x) $\vee$ femme(x))
Trouver les hommes qui ne sont père d'aucun enfant	{ x | homme(x) $\wedge$ $\neg$ $\exists$ y pere(x, y) }

4.1 – La représentation du sens en logique du premier ordre

1. Représenter le sens de chacune des phrases suivantes par une formule de la logique du premier ordre :
   a) *Tous les lions sont féroces.*
   b) *Quelques lions ne boivent pas.*
   c) *Aucun singe n'est soldat.*
   d) *Tous les singes sont malicieux.*
   e) *Tous les singes aiment une guenon.*

2. Représenter le sens de chacune des phrases suivantes par une formule de la logique du premier ordre, en utilisant les prédicats introduits en cours, plus les prédicats *frere*, *sœur* et *descendant*:
   a) *Jean Muller est frère d'Annie Muller*
   b) *Tout homme qui a un père a une mère.*
   c) *Jean Muller et Annie Muller ont les mêmes parents*
   d) *Si un individu a le même parent qu'un autre individu et s'il est un homme, alors il est frère du second.*

4.2 – Le principe de compositionnalité

- Le **principe de compositionnalité** dit que le sens littéral d'un énoncé est construit à partir du sens de ses parties et des règles de composition de ces parties.

- Appliqué à la phrase, il dit que le sens littéral d'une phrase est construit à partir du sens des mots qu'elle contient et de la façon dont ils se composent dans la syntaxe de la phrase.

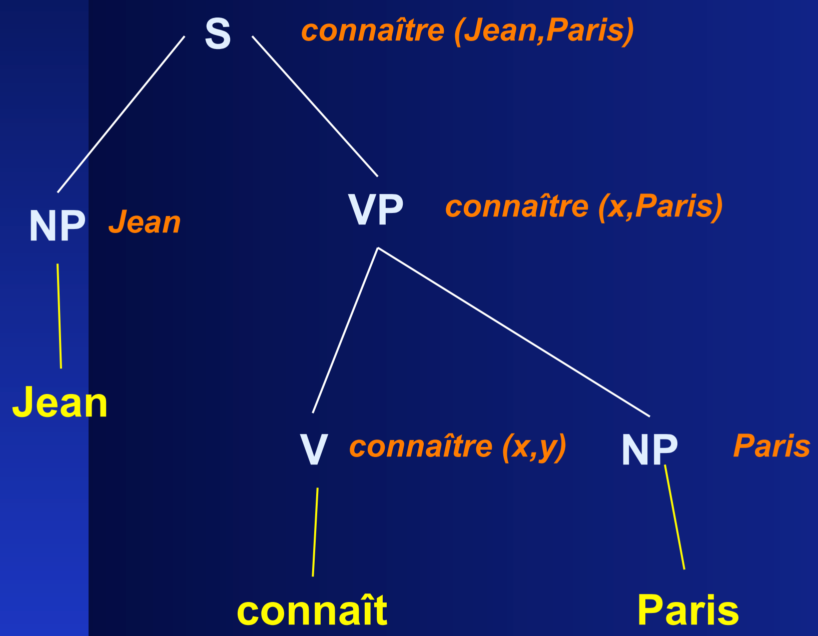
4.2 – Le principe de compositionnalité

- La conséquence est que l'entrée d'un **analyseur sémantique** de phrases peut être **l'arbre syntaxique** de ces phrases et un **lexique** qui donne une représentation du sens des mots.

- Il est nécessaire d'avoir aussi des règles qui permettent de composer le sens d'un nœud de l'arbre syntaxique à partir du sens de ses fils. Si l'arbre a été produit par une grammaire algébrique, l'idée est d'associer une **règle de composition sémantique** à chaque **règle syntaxique**.

4.2 – Le principe de compositionnalité

Analyse sémantique de la phrase : Jean connaît Paris



S → NP VP	*sem_{VP} [x := sem_{NP}]*
VP → V NP	*sem_{V} [y := sem_{NP}]*
NP → Jean	*Jean*
NP → Paris	*Paris*
V → connaît	*connaître (x,y)*

4.3 – Le lambda-calcul

- Le lambda-calcul est un formalisme inventé par Church en 1936 pour modéliser la notion de calcul à partir du concept de **fonction**.

- Les objets de base du lambda-calcul sont les **variables** et les **constantes**.

- A partir de ces objets de base, on construit des **lambda-termes** à l'aide des opérations d'**abstraction** et d'**application**.

- L'opération de **réduction** d'un lambda-terme permet de calculer sa valeur ou **forme normale.**

4.3 – Le lambda-calcul

- Définition formelle des lambda-termes construits sur un ensemble de variables et de constantes :

    - Toute **variable** ou **constante** est un lambda-terme.

    - Si M et N sont deux lambda-termes, l'**application** (M N) est un lambda-terme qui représente l'application de la fonction M à l'argument N.

    - Si M est un lambda-terme et x est une variable, l'**abstraction** $(\lambda x \,.\, M)$ est un lambda-terme représentant une fonction associant à toute valeur de x la valeur correspondante de M.

4.3 – Le lambda-calcul

- Si un lambda-terme contient un sous-terme de la forme (λx . M)N, ce sous-terme peut être contracté en M[x:= N], obtenu en remplaçant toutes les occurrences libres de x dans M par N. On dit que le lambda-terme global a été **réduit**.

- On peut itérer cette opération de réduction. Si l'itération termine, le résultat ne dépend pas de l'ordre des réductions et représente la **forme normale** du lambda-terme initial.

- Le processus de réduction d'un lambda-terme jusqu'à sa forme normale représente le calcul de la **valeur** de ce lambda-terme.

4.3 – Le lambda-calcul

- Lorsqu'on enchaîne des applications de gauche à droite, on peut omettre les parenthèses : $(\ldots((M_1\, M_2)\, M_3) \;\ldots\, M_n) \;=\; M_1\, M_2\, M_3 \;\ldots\, M_n$

- Dans le lambda-calcul, les fonctions de plusieurs variables sont représentées sous une forme **curryfiée :** une itération de fonctions à une seule variable ayant pour valeur une fonction.

  λ (x, y , z). f(x,y,z) est représentée par λ x. λ y. λ z. (((f x) y) z) ou sous forme simplifiée par λ x y z. f x y z

4.4 – L’analyse sémantique

- On utilise le lambda-calcul pour représenter la composition du sens d’une phrase à partir du sens de ses mots et des règles de composition syntaxique.

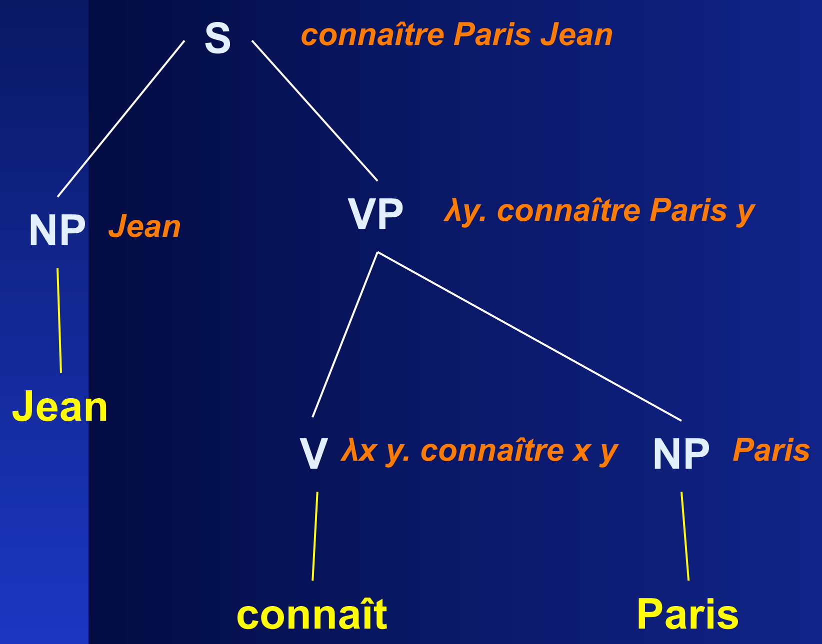
- Un lexique associe à chaque mot l’ensemble de ses représentations sémantiques sous forme de lambda-termes.

- A chaque règle de la grammaire algébrique, on associe une règle qui indique comment est construit le lambda-terme représentant le sens du non-terminal de sa partie gauche. Cette règle indique parmi les lambda-termes représentant le sens des non-terminaux, quelle est la fonction et dans quel ordre les arguments sont appliqués.

- Le sens d’une phrase est ensuite construit à partir de l’arbre syntaxique, pas à pas des feuilles jusqu’à la racine.

4.4 – L'analyse sémantique

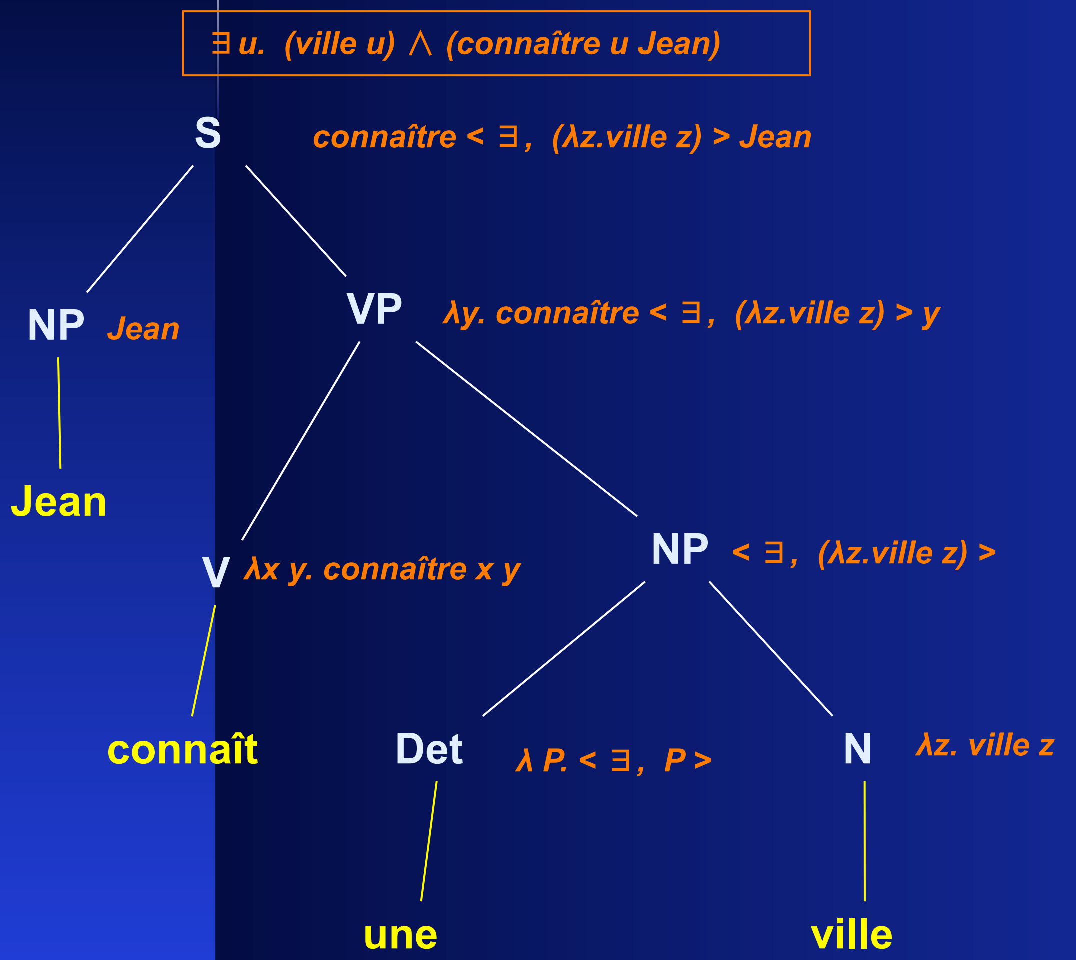
Analyse sémantique de la phrase : Jean connaît Paris



S → NP VP	$(sem_{VP}\ sem_{NP})$
VP → V NP	$(sem_{V}\ sem_{NP})$
NP → Jean	*Jean*
NP → Paris	*Paris*
V → connaît	*λx y. connaître x y*

4.4 – L'analyse sémantique

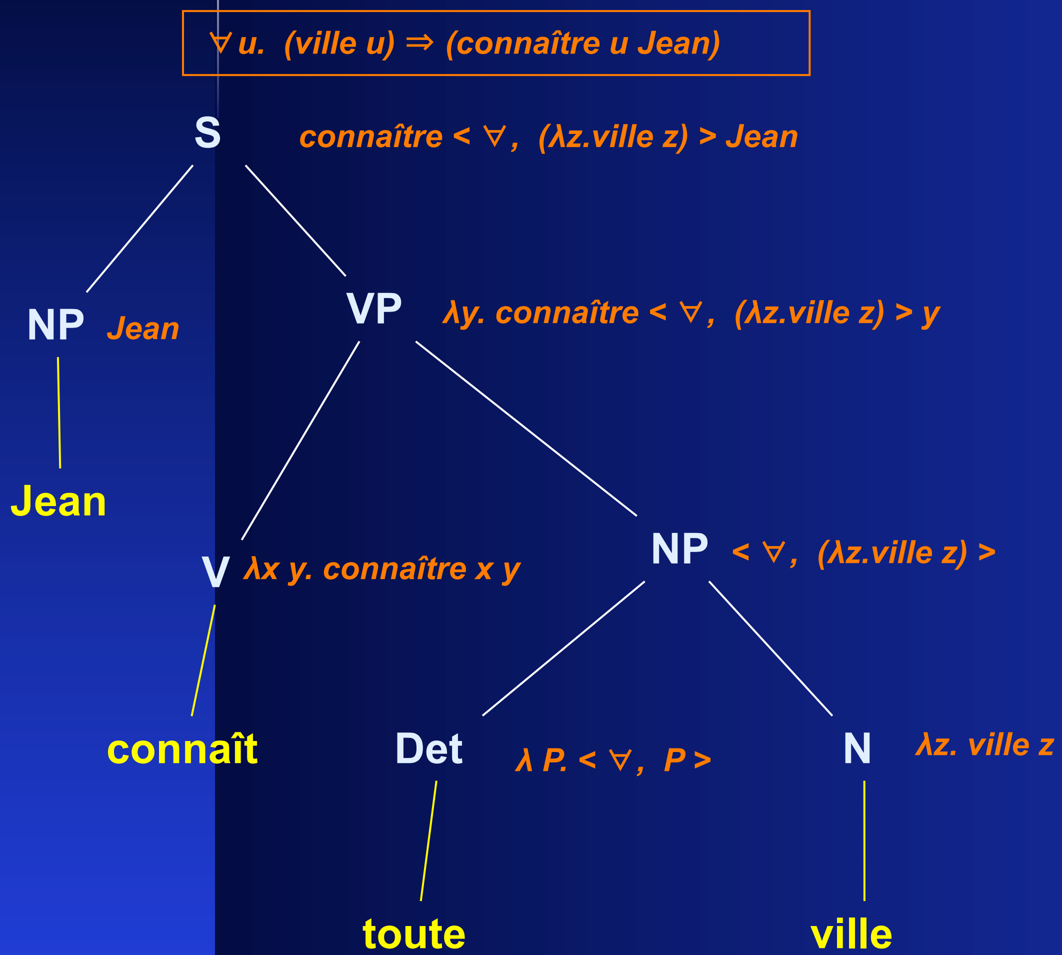
Analyse sémantique avec le quantificateur existentiel : Jean connaît une ville



S → NP VP	$(sem_{VP}\ sem_{NP})$
VP → V NP	$(sem_{V}\ sem_{NP})$
NP → Det N	$(sem_{Det}\ sem_{N})$
NP → Jean	*Jean*
Det → une	$\lambda P. < \exists, P >$
N → ville	$\lambda z.\ ville\ z$
V → connaît	$\lambda x\ y.\ connaître\ x\ y$

4.4 – L’analyse sémantique

Analyse sémantique avec le quantificateur universel : Jean connaît toute ville



S → NP VP	$(sem_{VP}\ sem_{NP})$
VP → V NP	$(sem_{V}\ sem_{NP})$
NP → Det N	$(sem_{Det}\ sem_{N})$
NP → Jean	*Jean*
Det → toute	*λ P. < ∀ , P >*
N → ville	*λz. ville z*
V → connaît	*λx y. connaître x y*

4.4 – L'analyse sémantique

1. Soit la grammaire algébrique G suivante :

$S \rightarrow NP\ \ VP \mid S \text{ et } S$

$NP \rightarrow Det\ \ N \mid NP\ PP \mid \text{marie} \mid \text{jean} \mid \text{pierre}$

$VP \rightarrow V1\ NP \mid V2\ NP\ PP \mid V3\ VP \mid VP \text{ et } VP \mid \text{arrive} \mid \text{s'asseoit} \mid \text{part}$

$V1 \rightarrow \text{connaît} \mid \text{connaître}$

$V2 \rightarrow \text{présente}$

$V3 \rightarrow \text{veut}$

$N \rightarrow \text{homme} \mid \text{élève} \mid \text{chanson} \mid \text{école}$

$PP \rightarrow Prep\ \ NP$

$Prep \rightarrow \text{à} \mid \text{d'}$

$Det \rightarrow \text{tout} \mid \text{un} \mid \text{une}$

4.4 – L'analyse sémantique

1. Faire l'analyse syntaxique des phrases suivantes avec la grammaire puis faire une analyse sémantique avec un lexique et une décoration des règles de la grammaire appropriés :
   a) *un homme arrive*
   b) *jean présente marie à pierre*
   c) *tout élève connaît jean.*
   d) *jean veut connaître marie.*
   e) *tout élève connaît un professeur*
   f) *jean arrive et marie part*
   g) *jean arrive et s'assoit*
   h) *tout élève d'une école connaît une chanson*