Sémantique véri-conditionnelle

.

- Connaître le sens d'une phrase, c'est connaître ses **conditions de vérité**, les contextes pour lesquels cette phrase est vraie.
- La phrase : *Jean a un nouveau vélo.*
- est vraie si Jean a un nouveau vélo.
- Distinction importante :
  - On ne sait pas si la phrase est vraie ou fausse.
  - Mais on sait déterminer dans quelles conditions elle est vraie ou fausse.
- Alfred Tarski 1935

Dénotation

.

- La **dénotation** d'une expression linguistique est l'objet du monde que cette expression désigne.
- La dénotation d'une expression α est notée $[[\alpha]]$.
- $[[\cdot]]$ est appelée **fonction d'interprétation**, elle permet d'associer, à une expression linguistique, des objets du monde.
- Elle permet d'établir un lien entre l'univers langagier et l'univers extra-linguistique.
- Plusieurs expressions linguistiques peuvent avoir la même dénotation.
  - *l'étoile du matin* , *l'étoile du berger*, *l'étoile du soir*, *Vénus*
  - *Napoléon*, *le vainqueur d'Austerlitz*, *le vaincu de Waterloo*.

Dénotation v/s Sens

.

- Certaines expressions de la langue n'ont pas de dénotation.
  - *le plus grand nombre entier*
  - *la quinzième planète du système solaire*
  - *la licorne de mon voisin*
- Pourtant ces expressions ne sont pas vides sémantiquement, elles sont parfaitement compréhensibles, et ce en vertu de leur **sens.**
- Le sens d'une expression est ce qui nous donne, ou ce qui nous permet de connaître, sa dénotation.

Dénotation v/s Sens

.

- Connaître le sens d’une phrase, c’est, si l’on connaît les circonstances (ie l’état du monde ou le modèle) auxquelles s’applique cette phrase, être capable de juger si la phrase est vraie ou fausse.
- Contrairement à la dénotation, le sens d’une expression est indépendant de la configuration du monde.
- *Expression linguistique* $\longrightarrow$ *Sens* $\longrightarrow$ *Dénotation*

Quelques dénotations

- La dénotation de la phrase *Jean dort* est une valeur de vérité, un élément de l'ensemble {vrai, faux}.
- La dénotation du nom propre *Jean* est l'individu Jean. On note D l'ensemble des individus du monde.
- La dénotation du verbe *dort* est une fonction de D vers {vrai, faux}.

Sens et logique

- Le sens de certaines expressions linguistiques peut être représenté par des formules de la logique du premier ordre.

Jean a un nouveau vélo

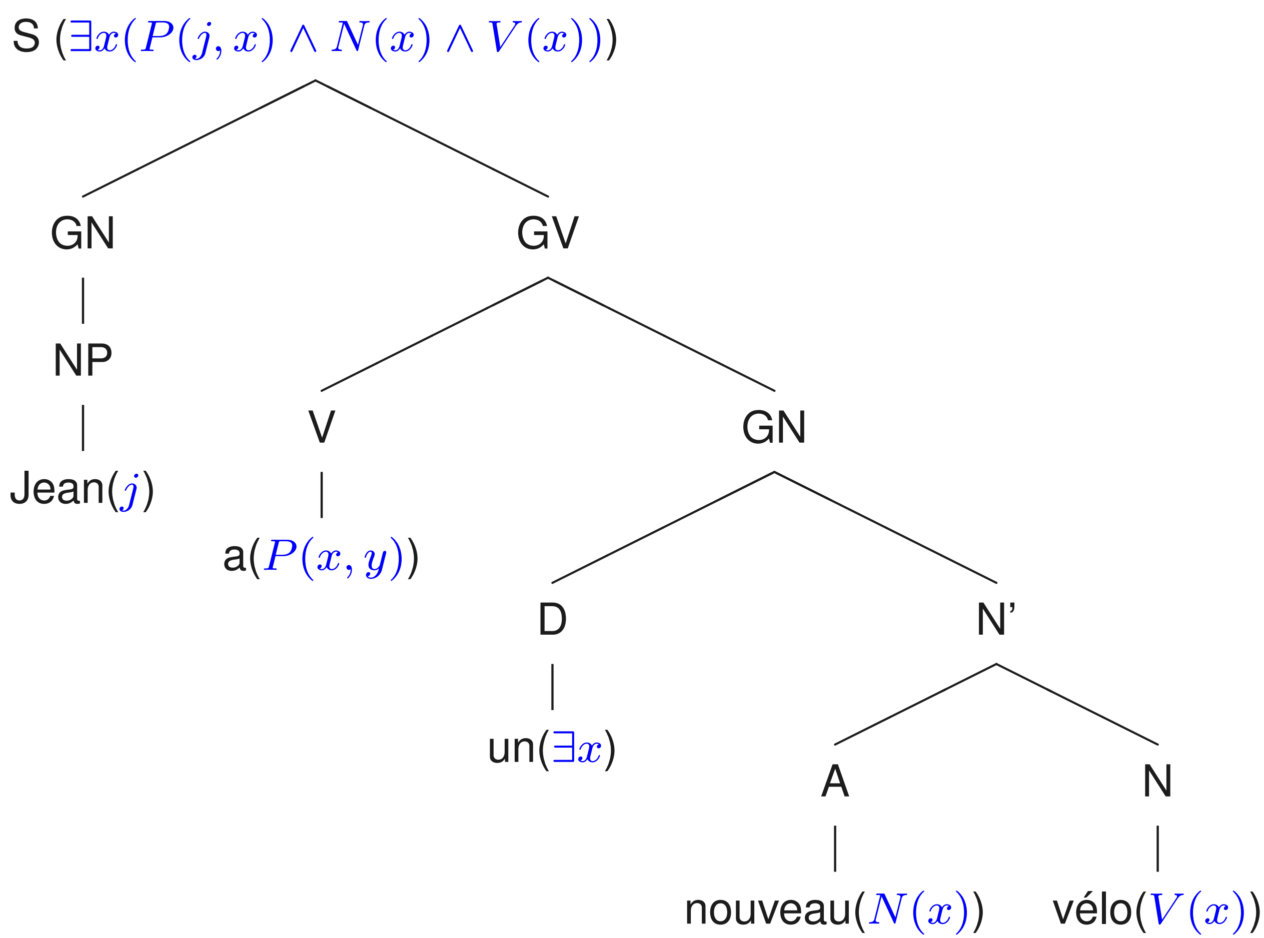
$$\exists x \quad (P(j, x) \quad \wedge \quad N(x) \quad \wedge \quad V(x))$$

- Principe de compositionnalité : *Le sens d'une expression complexe est une fonction du sens de ses parties et de leur mode de combinaison.*
- Les parties de l'expression sont déterminées par la syntaxe.

Construction du sens

.

- Comment construire la formule logique correspondant à une phrase à partir de l'analyse syntaxique de cette dernière ?



Comment combiner les sens ?

- Quelle opération permet de combiner les sens de différentes expressions ?
- La conjecture de Gottlob Frege : *And it is a natural conjecture that logical combination of parts into a whole is always a matter of saturating something unsaturated.*

sens saturés v/s non saturés

.

- Le sens de certains énoncés linguistiques se suffit à lui-même. Il est **saturé.**
  Exemples :
  - *César a conquis la Gaule.*
  - *César*
  - *la Gaule*
- Le sens de certains autres énoncés est **non saturé.**
  Exemple :
  - *a conquis la Gaule.*
- Il comporte une “place vide”, qui doit être occupée pour saturer son sens.

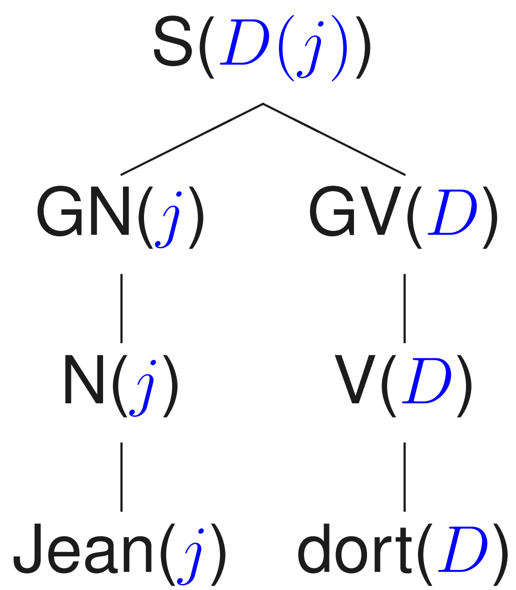
Parallèle avec les fonctions

.

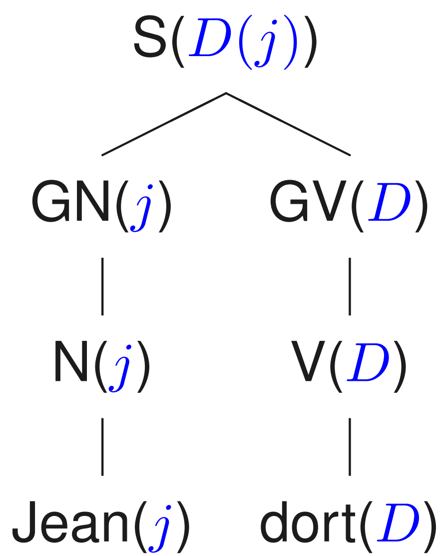
- Un sens non saturé peut être vu comme une fonction.
  - $Log(x)$
  - $a\_conquis\_la\_Gaule(x)$
- Pour que la fonction constitue une expression correcte, et que sa valeur puisse être calculée, elle a besoin d'un **argument** :
  - $Log(10)$
  - $a\_conquis\_la\_Gaule(César)$
- On dit que l'on **applique** la fonction à un élément, qui devient son argument.
- L'opération correspondante s'appelle l'**application fonctionnelle.**

Un premier exemple

.



Un premier exemple



- Lors d'une application fonctionnelle, comment distinguer l'argument de la fonction ?
- Pourquoi a t-on fait :
  - $D(j)$
  - et non $j(D)$?

Types sémantique

.

- On associe à toute expression un **type sémantique.**
- Le type sémantique d'une expression indique la nature de sa dénotation (un individu, un prédicat . . .).
- Un type sémantique permet de spécifier :
  - si le sens d'une expression est saturé ou pas
  - s'il ne l'est pas :
    - quelle est la nature de son argument
    - quelle est la nature de l'expression après application fonctionnelle
- Exemples de types sémantiques :
  - *Jean* est un élément de l'univers de discours.
  - *dort* est un prédicat unaire.

Types sémantiques

.

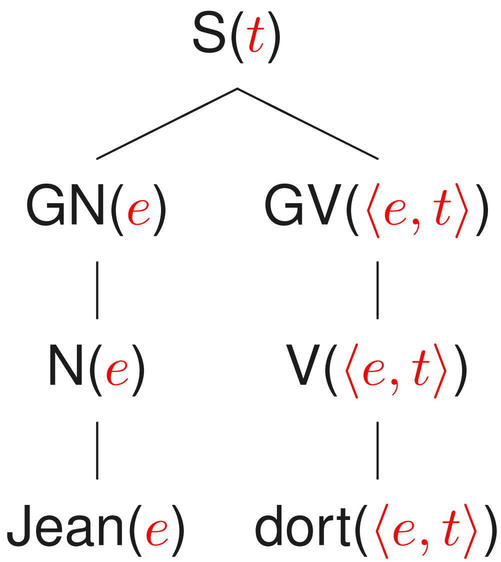
- Deux types saturés :
  - e les objets du monde
  - t les deux valeurs de vérité
- Des types non saturés de la forme :

$$\langle \underset{\text{type de l'argument}}{\alpha} , \underset{\substack{\text{type de l'expression} \\ \text{après application}}}{\beta} \rangle$$

- d'où la définition :
  - e et t sont des types sémantiques.
  - si α et β sont des types sémantique, alors $\langle \alpha, \beta \rangle$ sont des types sémantiques.
  - rien d'autre n'est un type sémantique.

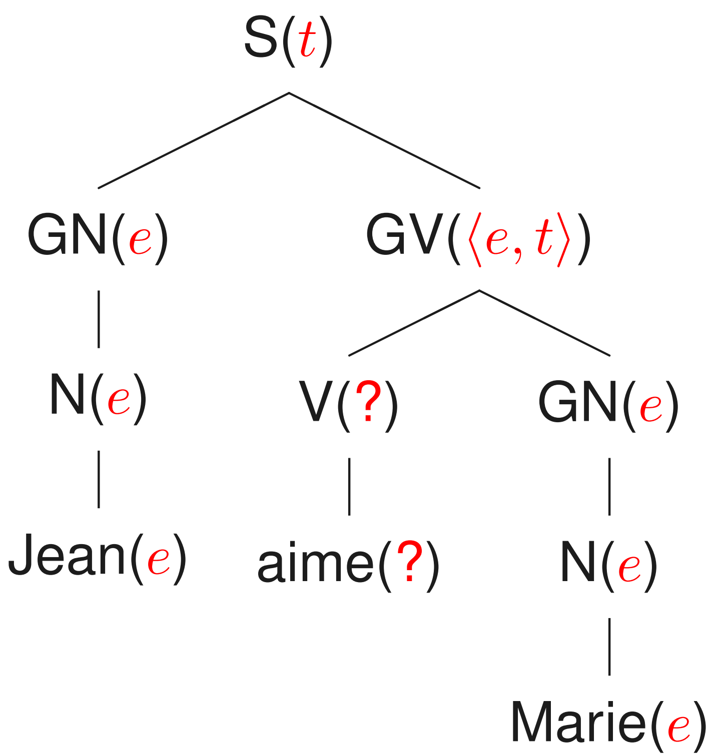
Types sémantiques

.



- Il n'est plus nécessaire d'indiquer lors d'une application fonctionnelle quel est la fonction et quel est l'argument.
- Le choix est guidé par les types sémantiques.

Un second exemple

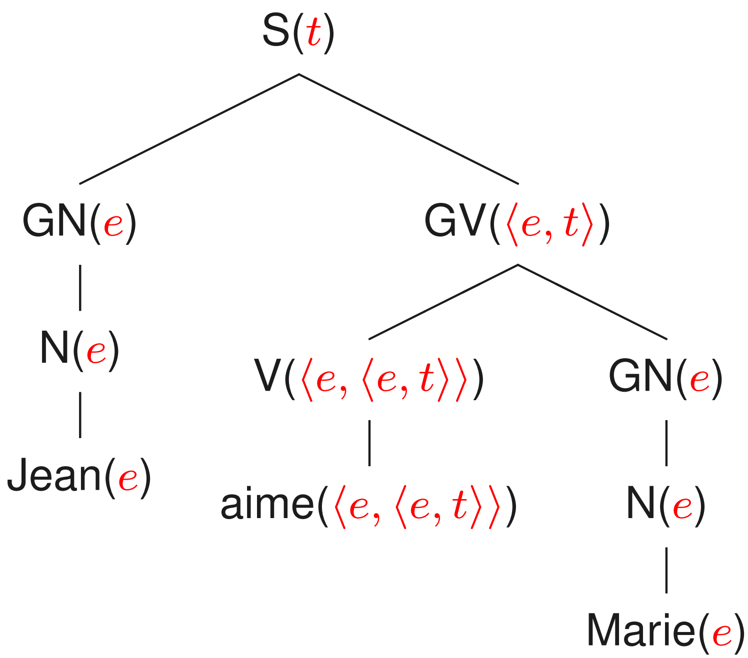


- Quel est le type de *aime* ?

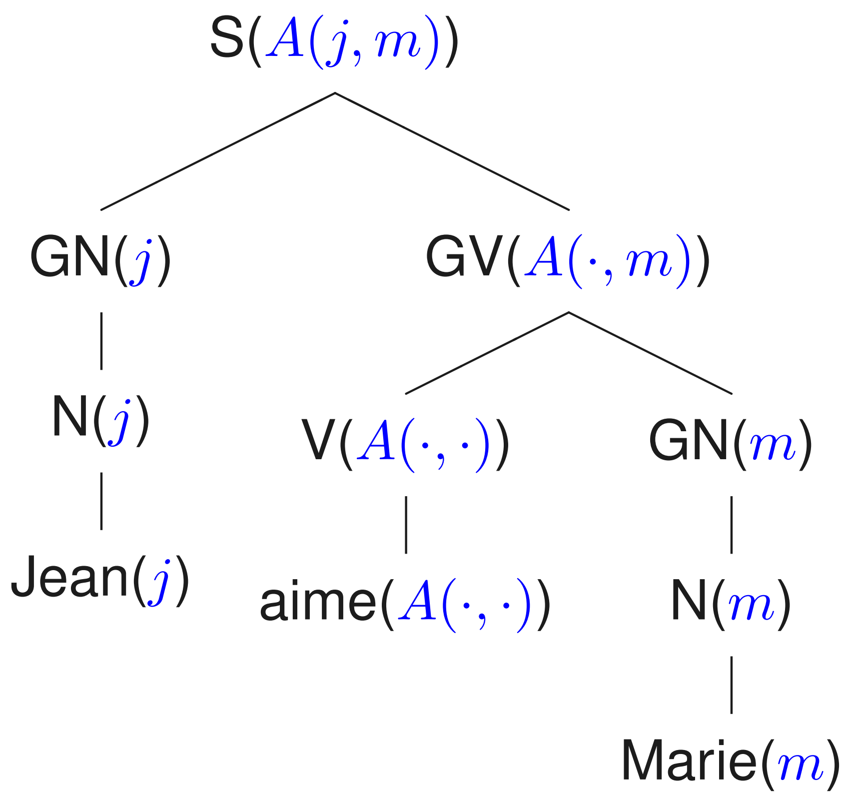
Un second exemple

.

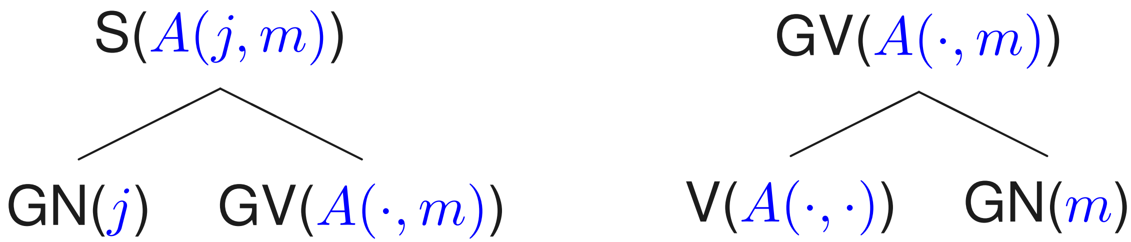
- *aime* prend un argument de type e et produit une expression de type $\langle e, t\rangle$
- Son type est donc : $\langle e, \langle e, t\rangle\rangle$



Un second exemple



Un problème technique



- Comment représenter le fait que
  - dans l'arbre de gauche j occupe la première position du prédicat A ?
  - dans l'arbre de droite m occupe la seconde position du prédicat A ?

Lambda calcul

.

- Le lambda calcul a été inventé permettre la construction d'expressions qui dénotent des fonctions de manière non ambiguë et de façon compositionnelle.
- La fonction $f(x) = x^2 + 3$ est décrite par l'expression :

$$\lambda x.x^2 + 3$$

- L'application de cette fonction à un argument z est décrite par l'expression :

$$(\lambda x.x^2 + 3)@(z)$$

Lambda calcul

.

- Le sens de dort peut être représentée par l'expression :

$$\lambda x.dort(x)$$

- Et son **application** à l'argument Jean par l'expression :

$$(\lambda x.dort(x))@(Jean)$$

- Comment passer
  - de $(\lambda x.dort(x))@(Jean)$
  - à $dort(Jean)$?

Lambda calcul

.

- Pour obtenir l'expression recherchée, il faut **substituer** *Jean* à la variable x.
- On note ${}_{[x \leftarrow t]}E$ la substitution du terme t à la variable x dans l'expression E.
- On a donc : ${}_{[x \leftarrow Jean]}dort(x) \equiv dort(Jean)$
- On définit la règle de β**-conversion** de la manière suivante :

$$(\lambda x.M)@(N) \stackrel{\beta}{\equiv}_{[x \leftarrow N]} M$$

- On a donc :

$$(\lambda {\color{red}x}.dort({\color{blue}x}))@({\color{red}Jean}) \stackrel{\beta}{\equiv} dort(Jean)$$

Lambda calcul - Exemples

.

$$((\lambda x.\lambda y.A(x,y))@(j))@(m) \stackrel{\beta}{\equiv} (\lambda y.A(j,y))@(m)$$
$$\stackrel{\beta}{\equiv} A(j,m)$$

$$(\lambda X.(X)@(j))@(\lambda x.H(x)) \stackrel{\beta}{\equiv} (\lambda x.H(x))@(j)$$
$$\stackrel{\beta}{\equiv} H(j)$$

$$((\lambda X.\lambda Y.(X)@(j) \wedge (Y)@(j))@(\lambda x.H(x)))@(\lambda x.G(x))$$
$$\stackrel{\beta}{\equiv} (\lambda Y.(\lambda x.H(x))@(j) \wedge (Y)@(j))@(\lambda x.G(x))$$
$$\stackrel{\beta}{\equiv} (\lambda Y.H(j)) \wedge (Y)@(j))@(\lambda x.G(x))$$
$$\stackrel{\beta}{\equiv} H(j) \wedge (\lambda x.G(x))@(j)$$
$$\stackrel{\beta}{\equiv} H(j) \wedge G(j)$$

Lambda calcul et types sémantiques

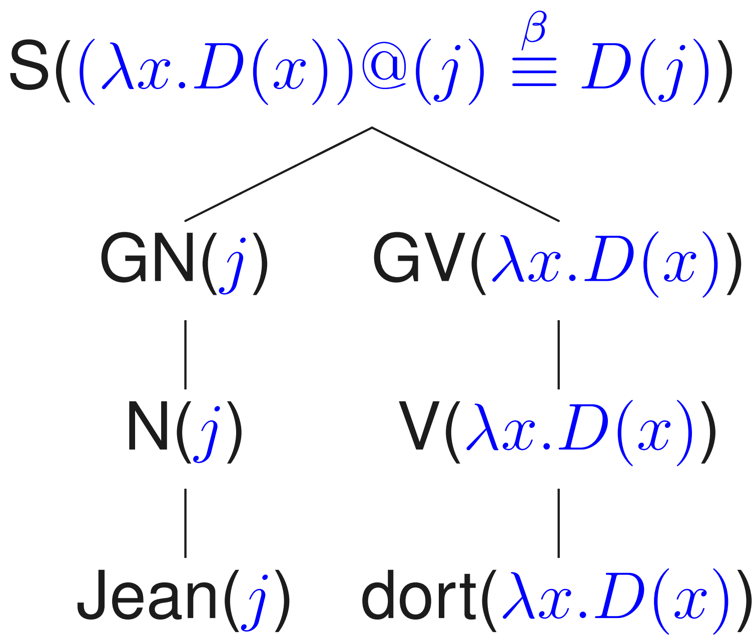
- Une application fonctionnelle entre deux lambda termes est valide si leurs types sont compatibles :

$$(\langle \alpha, \beta \rangle)@(\alpha)$$

-
$$\begin{array}{ccccc} (\lambda x.H(x)) & @ & (j) & \overset{\beta}{\equiv} & H(j) \\ \langle e, t \rangle & @ & e & \overset{\beta}{\equiv} & t \end{array}$$

Retour sur le premier exemple

.



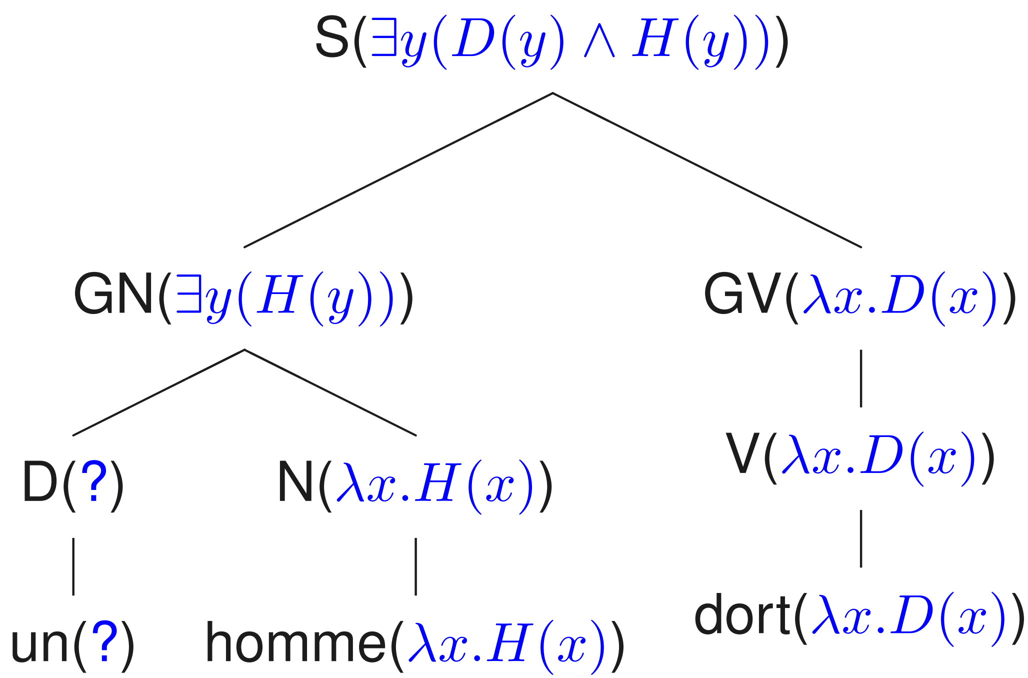
Retour sur le second exemple

- Le sens de *aime* est représentée par l'expression :

$$\lambda x.\lambda y.A(y,x)$$

- S($(\lambda y.A(y,m))@(j) \stackrel{\beta}{\equiv} A(j,m)$)
  - GN(j)
    - N(j)
      - Jean(j)
  - GV($(\lambda x.\lambda y.A(y,x))@(m) \stackrel{\beta}{\equiv} \lambda y.A(y,m)$)
    - V($\lambda x.\lambda y.A(y,x)$)
      - aime
    - GN(m)
      - N(m)
        - Marie(m)

Un troisième exemple



- Quelle est la contribution sémantique de *un* ?
  - Le quantificateur existentiel
  - et la structure générale de la formule

Un troisième exemple

.

S($\exists y(H(y) \wedge D(y))$)

GN($\lambda Y.\exists y(H(y) \wedge (Y)@(y))$) GV($\lambda x.D(x)$)

- Détails du calcul :

$$
\begin{array}{ll}
 & (\lambda Y.\exists y(H(y) \wedge (Y)@(y)))@(\lambda x.D(x)) \\
\overset{\beta}{\equiv} & \exists y(H(y) \wedge (\lambda x.D(x))@(y)) \\
\overset{\beta}{\equiv} & \exists y(H(y) \wedge D(y))
\end{array}
$$

Un troisième exemple

.

GN ($\lambda Y.\exists y(H(y) \wedge (Y)@(y))$)

D($\lambda X.\ \lambda Y.\exists y(\ (X)@(y)\ \wedge (Y)@(y))$) N($\lambda x.H(x)$)

$$(\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y)))@(\lambda x.H(x))$$

$$\equiv \lambda Y.\exists y(\lambda x.H(x))@(y) \wedge (Y)@(y))$$

$$\equiv \lambda Y.\exists y(H(y) \wedge (Y)y)$$

- Le sens de *un* est donc : $\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y))$
- *Il existe quelque chose qui est un X et qui Y.*
- Pour saturer le sens de *un*, il faut donner une valeur à X et à Y.
- Le type sémantique de *un* est donc : $\langle\langle e,t\rangle,\langle\langle e,t\rangle,t\rangle\rangle$

Un troisième exemple

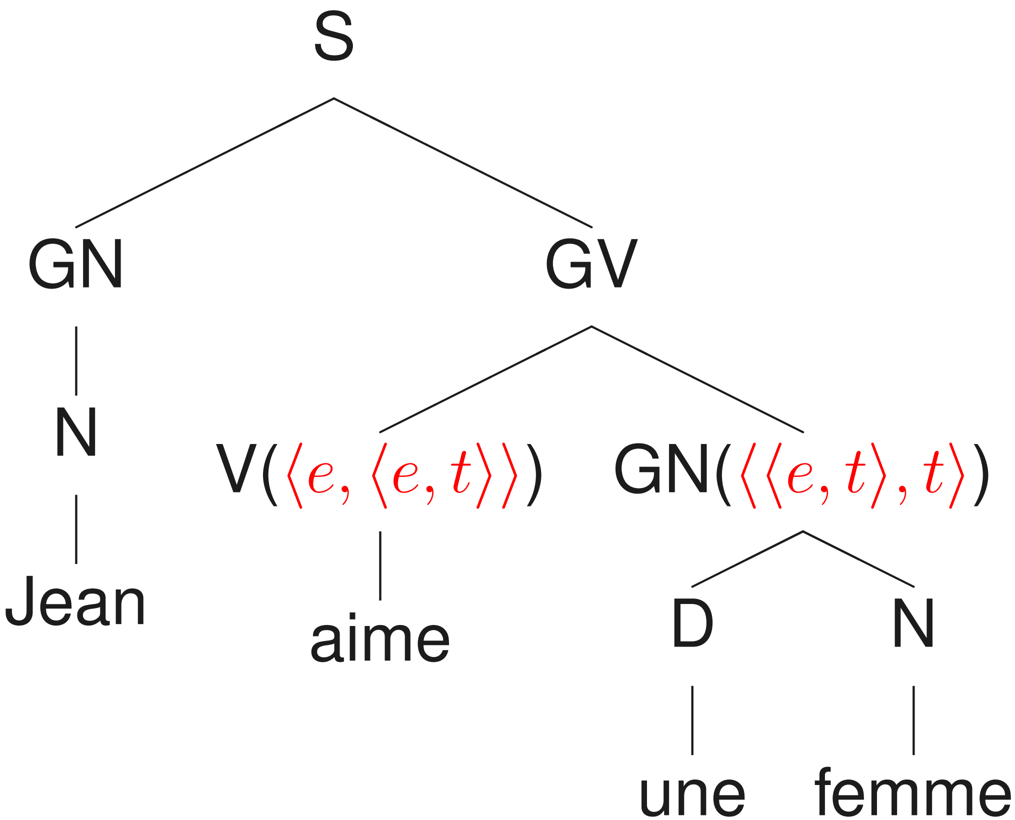
.

- S($\exists y(H(y) \wedge D(y))$)
  - GN($\lambda Y.\exists y(H(y) \wedge (Y)@(y))$)
    - D($\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y))$)
      - n($\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y))$)
    - N($\lambda x.H(x)$)
      - homme($\lambda x.H(x)$)
  - GV($\lambda x.D(x)$)
    - V($\lambda x.D(x)$)
      - dort($\lambda x.D(x)$)

Retour sur le verbe transitif

.

compatibilité des types du verbe et de l'objet direct.



Retour sur le verbe transitif

.

- On modifie le sens de *aime* :

$$\lambda y.\lambda x.A(x,y) \rightarrow \lambda X.\lambda x.(X)@(\lambda y.A(x,y))$$

- *aime une femme*

$$\begin{array}{ll} & (\lambda X.\lambda x.(X)@(\lambda y.A(x,y)))@(\lambda Y.\exists z(F(z) \wedge (Y)@(z))) \\ \stackrel{\beta}{\equiv} & \lambda x.(\lambda Y.\exists z(F(z) \wedge (Y)@(z)))@(\lambda y.A(x,y)) \\ \stackrel{\beta}{\equiv} & \lambda x.\exists z(F(z) \wedge (\lambda y.A(x,y))@(z)) \\ \stackrel{\beta}{\equiv} & \lambda x.\exists z(F(z) \wedge A(x,z)) \end{array}$$

- *aime* est du type : $\langle\langle\langle e,t\rangle,t\rangle,\langle e,t\rangle\rangle$

Retour sur le sens des noms propres

.

- La nouvelle définition de *aime* est incompatible avec un objet de type e.
- *aime Marie* : $(\langle\langle\langle e,t\rangle,t\rangle,\langle e,t\rangle\rangle)@(e)$
- On modifie le sens de *Marie* : $m \to \lambda X.(X)@(m)$.
- Le sens d'un nom propre est l'ensemble des propriétés possédées par le référent.
- Le nouveau sens est bien du type $\langle\langle e,t\rangle,t\rangle$.
- *aime Marie*

$$
\begin{array}{ll}
& (\lambda X.\lambda x.(X)@(\lambda y.A(x,y)))@(\lambda Y.(Y)@(m)) \\
\stackrel{\beta}{\equiv} & \lambda x.(\lambda Y.(Y)@(m))@(\lambda y.A(x,y)) \\
\stackrel{\beta}{\equiv} & \lambda x.(\lambda y.A(x,y))@(m) \\
\stackrel{\beta}{\equiv} & \lambda x.A(x,m)
\end{array}
$$

Adjectif épithète

.

N’($\lambda x.(V(x) \wedge N(x))$)

A(**?**) N($\lambda x.V(x)$)

nouveau(**?**) vélo($\lambda x.V(x)$)

- Quelle est la contribution sémantique de *nouveau* ?
  - Le prédicat N.
  - La conjonction.

Adjectif épithète

N'($\lambda x.(V(x) \wedge N(x))$)

A(λX $.\lambda x.(($ X $)@(x) \wedge N(x))$)

N($\lambda x.V(x)$)

nouveau

vélo($\lambda x.V(x)$)

- Détails du calcul :

$$
\begin{aligned}
& (\lambda X.\lambda x.((X)@(x) \wedge N(x)))@(\lambda y.V(y)) \\
\stackrel{\beta}{\equiv}\; & \lambda x((\lambda y V(y))@(x) \wedge N(x)) \\
\stackrel{\beta}{\equiv}\; & \lambda x(V(x) \wedge N(x))
\end{aligned}
$$

Retour sur la phrase initiale

GN $(\lambda Y.\exists y(V(y) \wedge N(y) \wedge (Y)@(y)))$

$(\lambda Y.\exists y(\lambda x(V(x) \wedge N(x))@(y) \wedge (Y)@(y)))$

$((\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y)))@(\lambda x(V(x) \wedge N(x))))$

D$(\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y)))$

n$(\lambda X.\lambda Y.\exists y((X)@(y) \wedge (Y)@(y)))$

N'$(\lambda x(V(x) \wedge N(x)))$

$(\lambda x.((\lambda x.V(x))@(x) \wedge N(x)))$

$((\lambda X.\lambda x.((X)@(x) \wedge N(x)))@(\lambda x.V(x)))$

A$(\lambda X.\lambda x.((X)@(x) \wedge N(x)))$

nouveau$(\lambda X.\lambda x.((X)@(x) \wedge N(x)))$

N$(\lambda x.V(x))$

vélo$(\lambda x.V(x))$

Retour sur la phrase initiale

S ($\exists y(V(y) \wedge N(y) \wedge P(j,y))$)

($(\lambda z.\exists y(V(y) \wedge N(y) \wedge P(z,y)))@(j)$)

($(\lambda Y.(Y)@(j))@(\lambda z.\exists y(V(y) \wedge N(y) \wedge P(z,y)))$)

GN($\lambda Y.(Y)@(j)$)

NP($\lambda Y.(Y)@(j)$)

ean($\lambda Y.(Y)@(j)$)

GV($\lambda z.\exists y(V(y) \wedge N(y) \wedge P(z,y))$)

($\lambda z.\exists y(V(y) \wedge N(y) \wedge (\lambda y.P(z,y)))@(y))$)

($\lambda z.(\lambda Y.\exists y(V(y) \wedge N(y) \wedge (Y)@(y)))@(\lambda y.P(z,y)))$)

($(\lambda Y.\lambda z.(Y)@(\lambda y.P(z,y)))@(\lambda Y.\exists y(V(y) \wedge N(y) \wedge (Y)@(y)))$)

V($\lambda Y.\lambda z.(Y)@(\lambda y.P(z,y))$)

GN ($\lambda Y.\exists y(V(y) \wedge N(y) \wedge (Y)@(y))$)

a($\lambda Y.\lambda z.(Y)@(\lambda y.P(z,y))$)

sources

.

- *Semantics in Generative Syntax* Irene Heim & Angelika Kratzer. *Blackwell Publishing*, 1998.
- *Mathematical Methods in Linguistics* Barbara Partee & Alice ter Meulen & Robert Wall *Springer*, 1990.
- Jacques Moeschler et Antoine Auchlin *Introduction à la linguistique contemporaine* Armand Colin, 2005.
- `http://www.semantique-gdr.net/dico/index.php/Accueil`
- *Sémantique linguistique et logique* Michel Galmiche *Presses Universitaires de France*, 1991.