Le rêve d'une formalisation purement fonctionnelle de la syntaxe et de la sémantique de la phrase :

l'approche catégorielle.

Historique (B. Wendling)

Husserl (cf. Frege)

syntaxe qui compose ...

... des catégories sémantiques !

opposition sujet / prédicat

nom / verbe

Ajdukiewicz ~ 1930

calcul non directionnel

$$n \quad \frac{S}{n\,n} \quad n \rightsquigarrow S$$

Pierre regarde marie

pas question d'ordre des mots/unités logique

Calcul de fractions

directionnel (Bar Hillel) 1953

Lambek calcul / logique
1958 / 1961 fonctions

Soyons un peu plus précis :

A Categorial Grammars

A.1. What are categorial grammars?

- A *lexicon* mapping words to (small) sets of formulas
- A *logic* specifying the meaning and the behaviour of the logical connectives

Universal grammar is a logic. Language variation is restricted to the lexicon.

A.2. AB grammars

Not a logic (yet!) but the foundation of categorial grammars.

A.3. Atomic formulas

groupes nominaux

s (sentence),

np (noun phrase), for example: John, the tall student

n (noun), for example: student, book, ...

Maybe some others: *pp* (for prepositional phrases), *inf* (for infinitival phrases), ...

Goal: all grammatical sentence should be derivable as being of category *s* (in a sense we will make precise).

A/B A sur B

$B\backslash A$ B sous A

A.4. Formulas

Formulas are inductively defined as follows.

- Atomic formulas are formulas. n, np, S
- If A and B are formulas, then (A/B) (we say A *over* B) and $(B\backslash A)$ (we say B *under* A) are formulas.

Intuition: a formula of the form A/B combines with a B to its *right* to form an A, a formula $B\backslash A$ combines with a B to its *left* to form an A.

A.5. Example formulas, example lexicon (strict)

The following are formulas: (np/n), $(np\backslash s)$, $((np\backslash s)/np)$, $((n\backslash n)/(np\backslash s))$

$((np\backslash s))/np$

$$
\begin{aligned}
Lex(the) &= \{\,(np/n)\,\}\\
Lex(an) &= \{\,(np/n)\,\}\\
Lex(president) &= \{\,n\,\}\\
Lex(actress) &= \{\,n\,\}\\
Lex(likes) &= \{\,((np\backslash s)/np)\,\}
\end{aligned}
$$

A.6. Example formulas, example lexicon (sloppy)

The following are formulas: np/n, $np\backslash s$, $(np\backslash s)/np$, $(n\backslash n)/(np\backslash s)$

$$
\begin{aligned}
Lex(the) &= np/n \\
Lex(an) &= np/n \\
Lex(president) &= n \\
Lex(actress) &= n \\
Lex(likes) &= (np\backslash s)/np
\end{aligned}
$$

A.7. AB grammars: rules

$$\frac{A/B \quad B}{A}\,[/E] \qquad\qquad \frac{B \quad B\backslash A}{A}\,[\backslash E]$$

A.8. AB grammars: rules

$$\frac{A/B \quad B}{A}\,[/E] \qquad \frac{\dfrac{\textit{the}}{np/n} \quad \dfrac{\textit{president}}{n}}{np}\,[/E]$$

$A = np$, $B = n$

president the

n np/n

? ?

A.9. AB grammars: rules

$$\frac{A/B \quad B}{A}\,[/E] \qquad \frac{\dfrac{\text{an}}{np/n} \quad \dfrac{\text{actress}}{n}}{np}\,[/E]$$

$A = np$, $B = n$

$$\frac{7\backslash 2 \quad 7}{??} \qquad \underbrace{2/7 \cdot 7}_{2}$$

A.10. AB grammars: rules

$$\frac{A/B \quad B}{A}\,[/E]$$

$$\dfrac{\dfrac{\begin{matrix}\text{the} & \text{president}\\ np/n & n\end{matrix}}{np} \qquad \dfrac{\dfrac{\textit{likes}}{(np\backslash s)/np} \quad \dfrac{\dfrac{\textit{an}}{np/n} \quad \dfrac{\textit{actress}}{n}}{np}\,[/E]}{np\backslash s}\,[/E]}{s}$$

$A = np\backslash s,\ B = np$

A.11. AB grammars: rules

$$\frac{B \quad B\backslash A}{A}\,[\backslash E] \qquad\qquad \begin{array}{c}\textit{likes an actress}\\ \vdots \\ np\backslash s\end{array}$$

$B = np$, $A = s$

A.12. AB grammars: rules

$$\frac{B \quad B\backslash A}{A}\,[\backslash E]$$

the president likes an actress

$$\frac{\overset{\vdots}{np} \qquad\qquad \overset{\vdots}{np\backslash s}}{s}\,[\backslash E]$$

$B = np$, $A = s$

A.13. Modifiers

chat noir

$$\frac{X \qquad X\backslash X}{X}$$

1. A student slept.
2. A student slept in class.
3. A student slept in class during the exam.
4. A student slept in class during the exam yesterday at 15h while snoring.

"*in class*" modifies a sentence s and is therefore assigned the formula $s\backslash s$ (or if you prefer, the vp modifier $(np\backslash s)\backslash(np\backslash s)$).

"*class*" is a noun n, therefore a lexical possibility for "*in*" should be $(s\backslash s)/n$ or $((np\backslash s)\backslash(np\backslash s))/n$.

A.14. Relative phrases

The	student	who	slept
np/n	n	$(n\backslash n)/(np\backslash s)$	$np\backslash s$

The	student	whom	the professor	woke
np/n	n	$(n\backslash n)/(s/np)$	np	$(np\backslash s)/np$

AB G | CFG

chat : n | $n \to$ chat

$\underbrace{b/a \quad a \to b}$ | $b \to b/a \quad a$

$\to$ pour les a, b qui apparaissent dans le lexique

A.15. Context free grammars

A context-free grammar is defined by:

[Non Terminals] a set NT of symbols called non terminals, one of them being the start symbol S.

[Terminals] set T of symbols, disjoint from NT, called terminals (or words according to the linguistic viewpoint)

[Production rules] a finite set of production rules of the form $X \to W$ with $X \in NT$ and $W \in (T \cup NT)^*$

~~S → phrase~~]

A.16. Context free grammars

A context free grammars is said to be:

- in strong Greibach normal form when all rules are $X \to a$ or $X \to aY$ or $X \to aYZ$, with $a \in T$ and $X, Y, Z \in NT$
- in Chomsky normal form when all rules are $X \to a$ or $X \to YZ$ with $a \in T$ and $X, Y, Z \in NT$

Any context free grammar can be turned into an a grammar of both normal forms, both generating the same language.

A.17. From AB grammars to context free grammars

Given an AB grammar, there exists a context free grammar (in Chomsky normal form) that generates the same language.

Take all categories and subcategories from the lexicon as non terminals, add rules:

$Y \to X\ (X \backslash Y)$,

$Y \to (Y/X)\ X$ and

$X \to a$ whenever $a : X$

A.18. From context free grammars to AB grammars

Given a context free grammar, which can be assumed to be in Greibach normal form, there exists an AB grammar that generates the same language.

CFG AB G

$X \to a$ becomes $a : X$

$X \to aY$ becomes $a : X/Y$

$X \to aYZ$ becomes $a : (X/Z)/Y$

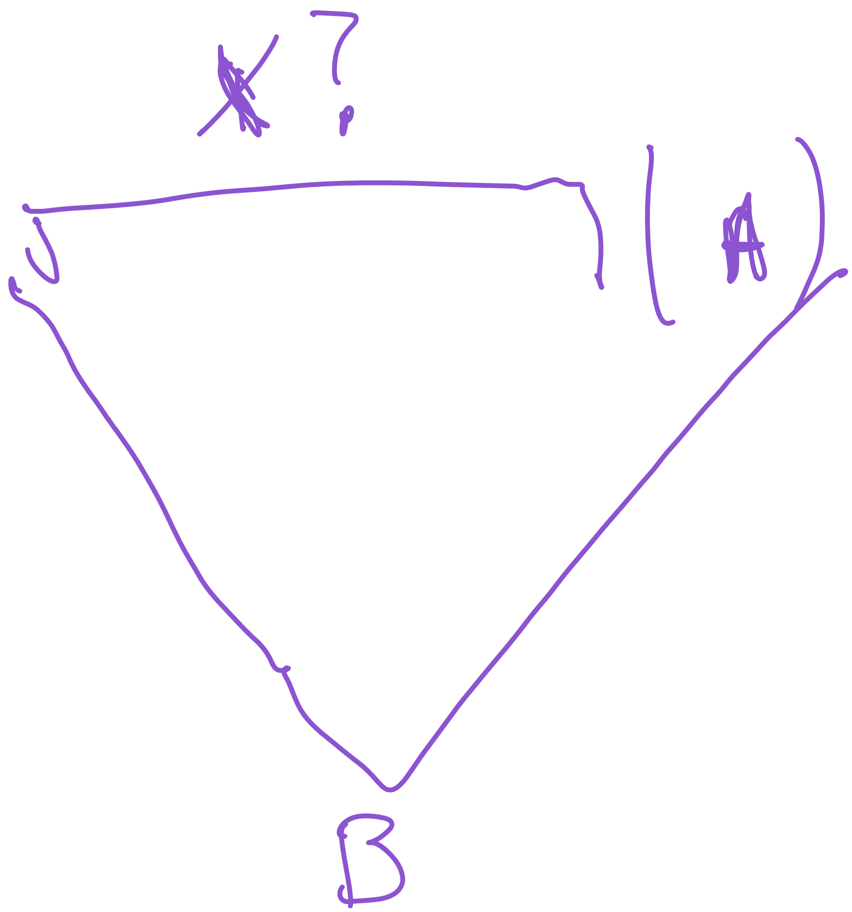
un seul /

A.19. Lambek grammars — Natural deduction

$$\frac{A/B \quad B}{A}\,[/E] \qquad\qquad \frac{\begin{array}{r}\dots[B]^n\\ \vdots\ \ \\ A\ \ \end{array}}{A/B}\,[/I]^n$$

$$\frac{B \quad B\backslash A}{A}\,[\backslash E] \qquad\qquad \frac{\begin{array}{l}[B]^n\dots\\ \ \ \vdots\\ \ \ A\end{array}}{B\backslash A}\,[\backslash I]^n$$

Conditions: $[B]$ is the rightmost (for $/I$) resp. leftmost (for $\backslash I$) undischarged hypothesis *and* the proof has another undischarged hypothesis. $[B]$ is discharged after application of the rule.



$$X = B/A$$

$$\frac{B/A \quad A}{B}$$

A.20. A very interesting book

a	*very*	*interesting*	*book*
np/n	$(n/n)/(n/n)$	n/n	n

n/n

n

np

A.21. *A very book

un exemple très

$(n\backslash n)/(n\backslash n)$ $\quad$ (ε) $\cancel{n}$ $n\backslash n$

voici

S/np $\quad np/n \quad n \quad n\backslash n$

n

np

S

a $\quad$ *very* $\quad$ n *book*

$np/n \quad (n/n)/(n/n) \; n\backslash n \quad n$

Voici un exemple très simple

$S/np \quad np/n \quad n \quad (n\backslash n)/(n\backslash n) \quad n\backslash n$

$n\backslash n$

n

np

S

A.23. Relatives

np

The	*student*	*whom*	*the professor*	*woke*	
np/n	n	$(n\backslash n)/(s/np)$	np	$(np\backslash s)/np$	$\cancel{np}$

$np\backslash s$

s/n

n/n

np trace ≈ hypothetical *np*

movement ≈ introduction rule

Fonctions !

$f : A \to B$

A A B $\rightsquigarrow$ B f : fonction B/A qui transforme un A à droite en un B

$f : B \leftarrow A$

B A A

$$\dfrac{\dfrac{X \quad A}{B}}{X : B/A}$$

Lambek

si X transforme un A à droite en un B alors X est un B/A

une fonction de A dans B

Limites et problèmes

(~~spurious~~ ambiguities
→ normal proofs)

constituants discontinus
extraposition
ordre relativement libre

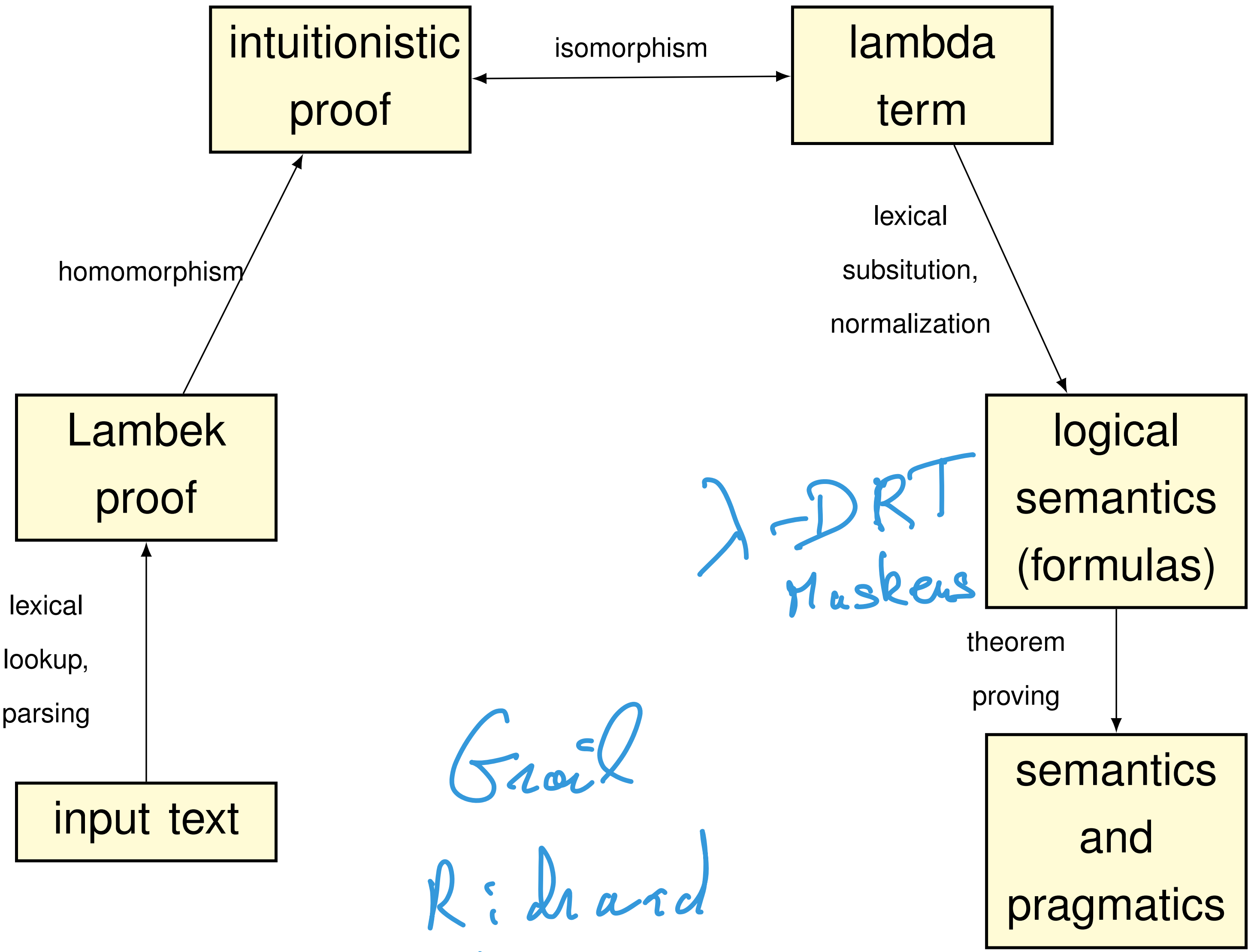
MMCG
ou
CCG

C Montague Semantics

C.1. Overview

- Montague Grammar and the simply typed lambda calculus (reminder)
- Curry-Howard formulas-as-types interpretation
- Montague semantics for the Lambek calculus

C.2. Architecture



λ-DRT Muskens

Gaël Richard MOOT

	Introduction rules	Elimination rules
Intuitionistic	$\dfrac{\begin{array}{c}[A]^n \\ \vdots \\ B\end{array}}{A \to B} \to I_n$	$\dfrac{A \quad A \to B}{B} \to E$
Lambek	$\dfrac{\begin{array}{c}[A]^n \ldots \\ \vdots \\ B\end{array}}{A \backslash B} \backslash I_n$ $\dfrac{\begin{array}{c}\ldots [A]^n \\ \vdots \\ B\end{array}}{B / A} / I_n$	$\dfrac{A \quad A \backslash B}{B} \backslash E$ $\dfrac{B / A \quad A}{B} / E$

C.3. Types and terms: Curry-Howard

Brouwer Heyting Kolmogorov

A proof of $A \to B$ is a function that maps proofs of A to proofs of B.

Think of a formula/type as the set of its proofs.

Types are.... formulae.

λ-terms encode proofs $u : U$ means u is a term of type U.

We will also write $u : U$ as u^U.

$(\lambda x^A \; u^B) : A \to B$

C.4. Terms: Curry-Howard

$f : \mathbb{N} \to \mathbb{N}$

$p \mapsto 2p+1$

$\boxed{f} : A \to B$

$x \mapsto u$

1. *hypotheses* variables of each type which are terms of this type
2. *constants* there can be constants of each type
3. *abstraction* if $x : U$ is a **variable** and $t : T$ then $(\lambda x^U.\, t) : U \to V$.
4. *application* if $f : U \to V$ and $t : U$ then $(f\ t) : V$

With such typed terms we can faithfully encode proofs.

Variables are hypotheses (that are simultaneously cancelled).

C.5. Reduction and Normalisation

$f : \mathbb{N} \to \mathbb{N}$

$p \to 2p+1$

$f(a-2b)$

$2\underbrace{(a-2b)}_{p}+1$

Reduction: $(\lambda x : U.\ t)^{U \to V} u^U$ reduces to $t[x := u] : V$.

Every simply typed lambda term reduces to a unique normal form, regardless the reduction strategy used.

C.6. Representing formulae within lambda calculus — connectives

church

e: entities ι
t: truthvalues o
prop

FOL ∀x shaves(x,x)

Assume that the base types are **e** and **t** and that the only constants are

We need the following logical constants:

e entities ι
individual

t truth values
prop

Constant	Type
$\exists$	$(e \to t) \to t$
$\forall$	$(e \to t) \to t$
$\wedge$	$t \to (t \to t)$
$\vee$	$t \to (t \to t)$
$\supset$	$t \to (t \to t)$

$a \to (b \to c) = a\, b \to c$

$(a \to b) \to c$

fonction de fonction

Church: FOL formulas as simply typed λ-terms

C.7. Representing formulae within lambda calculus — language constants

The language constants for First Order Logic (for a start):

dort : e -> t
regarde : e -> e -> t

- R_q of type $\mathbf{e} \to (\mathbf{e} \to (.... \to \mathbf{e} \to \mathbf{t}))$
  e.g. likes: $e \to e \to t$, sleeps $e \to t$
- f_q of type $\mathbf{e} \to (\mathbf{e} \to (.... \to \mathbf{e} \to \mathbf{e}))$

C.8. Formulae and normal lambda terms

Proposition 4 *A normal lambda-term of type* $\mathbf{t}$ *using only the constants given above corresponds to a formula of first-order logic.*

C.9. Example: From formulae to normal lambda terms

$\forall x. barber(x) \supset shaves(x, x)$

$\exists x$

$\exists$

$\cancel{\forall}(\lambda x^{\mathbf{e}}. (\supset barber(x))((shaves(x))(x)))$

prefix

Another one?

Detailed examples: a FOL formula as a term and as a natural deduction proof.

C.10. For Montague semantics

Non normal lambda terms of type $\mathbf{t}$ coming from syntax do not really correspond to formulae.

Hence we need:

- normalisation
- a proof that the normal terms do correspond to formulae, as we just shown.

C.11. Montague semantics. Types.

Simply typed lambda terms

$$types ::= e \mid t \mid types \to types$$

chair , *sleep* $e \to t$

likes transitive verb $e \to (e \to t)$

C.12. Montague semantics: Syntax/semantics.

(Syntactic type)*	=	Semantic type	
s^*	=	t propositions	a sentence is a proposition
groupes nominaux np^*	=	e entités	a noun phrase is an entity
nom commun n^*	=	$e \rightarrow t$ prédicat	a noun is a subset of the set of entities
$\underbrace{(A\backslash B)^* = (B/A)^*}_{\text{Bar Hillel / Lambek}}$	=	$\underbrace{A \rightarrow B}_{\text{Husserl}}$	extends easily to all syntactic categories of a Categorial Grammar e.g. a Lambek CG

Logical operations (and, or, some, all the,.....) are the lambda-term constants defined above.

C.13. Montague semantics
Logic within lambda-calculus

Words in the lexicon need constants for their denotation:

likes	$\lambda x \lambda y\ (\text{likes}\ y)\ x$	$x : e,\ y : e, \text{likes} : e \rightarrow (e \rightarrow t)$
`<< likes >> is a two-place predicate`		
Garance	$\lambda P\ (P\ \text{Garance})$	$P : e \rightarrow t,\ \text{Garance} : e$
`<< Garance >> is viewed as the properties that << Garance >> holds`		

$(e \rightarrow t) \rightarrow t$ type raising de e

C.14. Montague semantics. Computing the semantics 1/5

1. Replace in the lambda-term issued from the syntax the words by the corresponding term of the lexicon.
2. Reduce the resulting λ-term of type t to obtain its normal form, which corresponds to a logical formula, the "meaning".

np $\quad e$

$n \quad e \to t$

$s \quad t$

word	***syntactic type*** u ***semantic type*** u^* ***semantics: λ-term of type*** u^* x^v ***means that the variable or constant*** x ***is o***
some	$(s/(np\backslash s))/n$ grammar
	$(e \to t) \to ((e \to t) \to t)$
	meaning $\lambda P^{e\to t}\ \lambda Q^{e\to t}\ (\exists^{(e\to t)\to t}\ (\lambda x^e(\wedge^{t\to(t\to t)}\underbrace{(P\ x)}_{1}\underbrace{(Q\ x)}_{2}$
statements	n
	$e \to t$
	$\lambda x^e(\mathtt{statement}^{e\to t}\ x)$
speak_about	$(np\backslash s)/np$
	$e \to (e \to t)$
	$\lambda y^e\ \lambda x^e\ ((\mathtt{speak\_about}^{e\to(e\to t)}\ x)\ y)$
themselves	vt $((np\backslash s)/np)\backslash(np\backslash s)$ vi
	pred binaire $(e \to (e \to t)) \to (e \to t)$ predicat-unaire
	$\lambda P^{e\to(e\to t)}\ \lambda x^e\ ((P\ x)\ x)$

C.15. Syntactic proof

Let us first show that "*Some statements speak about themselves*" belongs to the language generated by this lexicon. So let us prove (in natural deduction) the following:

$$(s/(np\backslash s))/n \; , \; n \; , \; (np\backslash s)/np \; , \; ((np\backslash s)/np)\backslash(np\backslash s) \vdash s$$

some statement speakabout themselves

$$\dfrac{\dfrac{(s/(np\backslash s))/n \qquad n}{(s/(np\backslash s))}\,/E \qquad \dfrac{(np\backslash s)/np \qquad ((np\backslash s)/np)\backslash(np\backslash s)}{(np\backslash s)}\,\backslash E}{s}\,/E$$

C.16. Syntactic Proof to Semantic proof

$n \quad e \to t$
$np \quad e$
$s \quad t$

$$\dfrac{\dfrac{(s/(np\backslash s))/n \quad n}{(s/(np\backslash s))}/E \qquad \dfrac{(np\backslash s)/np \quad ((np\backslash s)/np)\backslash(np\backslash s)}{(np\backslash s)}\backslash E}{s}/E$$

Using the homomorphism from syntactic types to semantic types we obtain the following intuitionistic deduction.

$$\dfrac{\dfrac{(e \to t) \to (e \to t) \to t \quad e \to t}{(e \to t) \to t}\to E \qquad \dfrac{e \to e \to t \quad (e \to e \to t) \to e \to t}{e \to t}\to E}{t}\to E$$

C.17. Semantic Proof to Lambda Term

insert λ terms from the ~~lex~~lexicon

$$\dfrac{\dfrac{(e\to t)\to(e\to t)\to t \qquad e\to t}{(e\to t)\to t}\to E \qquad \dfrac{e\to e\to t \qquad (e\to e\to t)\to e\to t}{e\to t}\to E}{t}\to E$$

$$\dfrac{\dfrac{So^{(e\to t)\to(e\to t)\to t} \qquad Sta^{e\to t}}{(So\ \ Sta)^{(e\to t)\to t}}\to E \qquad \dfrac{SpA^{e\to e\to t} \qquad Refl^{(e\to e\to t)\to e\to t}}{(Refl\ \ SpA)^{e\to t}}\to E}{((So\ \ Sta)\ \ (Refl\ \ SpA))^{t}}\to E$$

proposition

C.18. Montague semantics. Computing the semantics. 3/5

The syntax (e.g. a Lambek categorial grammar) yields a λ-term representing this deduction simply is

((some statements) (themselves speak_about)) of type t

C.19. Montague semantics. Computing the semantics. 4/5

$$\Big(\big(\lambda P^{e\to t}\,\lambda Q^{e\to t}\,(\exists^{(e\to t)\to t}\,(\lambda x^e(\wedge(P\;x)(Q\;x))))\big)$$
$$(\lambda x^e(\mathtt{statement}^{e\to t}\;x))\Big)$$
$$\Big(\big(\lambda P^{e\to(e\to t)}\,\lambda x^e\,((P\;x)x)\big)$$
$$(\lambda y^e\,\lambda x^e\,((\mathtt{speak\_about}^{e\to(e\to t)}\;x)y))\Big)$$

$$\downarrow\beta$$

$$(\lambda Q^{e\to t}\,(\exists^{(e\to t)\to t}\,(\lambda x^e(\wedge^{t\to(t\to t)}(\mathtt{statement}^{e\to t}\;x)(Q\;x)))))$$
$$(\lambda x^e\,((\mathtt{speak\_about}^{e\to(e\to t)}\;x)x))$$

$$\downarrow\beta$$

$$(\exists^{(e\to t)\to t}\,(\lambda x^e(\wedge(\mathtt{statement}^{e\to t}\;x)((\mathtt{speak\_about}^{e\to(e\to t)}\;x)x))))$$

formule car λ terme normal de type t

C.20. Montague semantics. Computing the semantics. 5/5

This term represent the following formula of predicate calculus (in a more pleasant format):

formula:

$$\exists x : e\ (\mathtt{statement}(x) \land \mathtt{speak\_about}(x, x))$$

λ-term: proof of the correctness of the formula

This is a (simplistic) semantic representation of the analysed sentence.

Tout est fonction

dans l'écriture des formules.

On distingue bien

la formule

de sa réalité

Limites et problèmes

- sens lexical ?? → MGL
- ambiguités de portée
  → plusieurs analyse

Formalisme pour la syntaxe et la sémantique
Grammaires de Lambek + sémantique de Montague

\+ simple / élégant
  - peu de règles
  - structures simples
  - lexicalisé

\+ bonnes propriétés

\- limites en particulier syntaxiques
  et pb de compositionnalité (DRT)

\- pratiques : complexité, données, ...

Limites syntaxiques ....

- CCG Steedman (règles non logiques)
  ce n'est plus un système logique
  on perd la transparence synt / sém

- codage de LTAG (Pogodalla, Lecomte, Stabler)
  des grammaires minimalistes
  codage compliqué
  lien moins clair avec la sémantique

- MMCG système logique
  "bizarre" mais logique
  pas de pb avec la sémantique modalité transparente

- acquisition du lexique syntaxique

machine learning

- complexité

machine learning

+ vérification

Côté sémantique (plus facile)

- λ DRT pour les pb de non compositionalité
- sémantique lexicale en théorie des types
- complexité ? Soft LL (polynomiale)
- acquisition du lexique sémantique
  Montague OK
  sémantique lexicale ~~$~~ compliqué !

Pour les dérivations (preuves formelles dans des logiques exotiques)

graphes (algorithmiques plus facile)

Proof nets Réseaux

In practice **PROOF NETS**

implementation of categorial grammars

- more efficient for proof search (machine learning step)

GRAIL Richard MOOT

<u>actually</u>

- extension of Lambek grammars: (word order ... $\longrightarrow$ MMCG)
- semantic λ-DRT FOL formulas + handling (references pronouns)

GRAIL categorial parser

syntactic (categorial style)

semantic (Montague style)

wide coverage

Step 1

grammar acquisition

semantic λ terms?
just for the logical structure
(grammatical words)
otherwise "chain" $\lambda x e$ chain(x) $e \to t$

Un corpus de référence pour le français

Une ressource lexicale et syntaxique richement annotée (et validée manuellement) pour les linguistes, utilisable en TAL.

- Projet initié en 1997, avec le soutien de l'IUF, du CNRS et du CNRTL
- 21 550 phrases (environ 664 500 tokens) du journal *Le Monde* (1990-1993)
- métadonnées : auteur, date, domaine (par article)
- Annotations lexicales (catégories, sous-catégories, flexion, mots composés avec composants) et syntaxiques (constituants majeurs, fonctions grammaticales) validées
- Corpus annoté téléchargeable (version 1.0 2016) en plusieurs formats (xml, Tiger-xml, PTB, CoNLL)

La diminution paraît, toutefois, moins nette en France et en Italie.

Sélectionnez le format de sortie

Texte | XML | PTB | Tiger | CoNLL

```
(SENT (NP-SUJ (D La) (N diminution)) (VN (V paraît)) (PONCT ,) (ADV toutefois) (PONCT ,) (A
```

Visualisation graphique

step 2

input → supertagging (deep learning step)

categoric combre

reason!

Too many categories per word

W_1 $\qquad$ W_{10} $\qquad$ 10^{10}

10 $\qquad$ 10

step 3

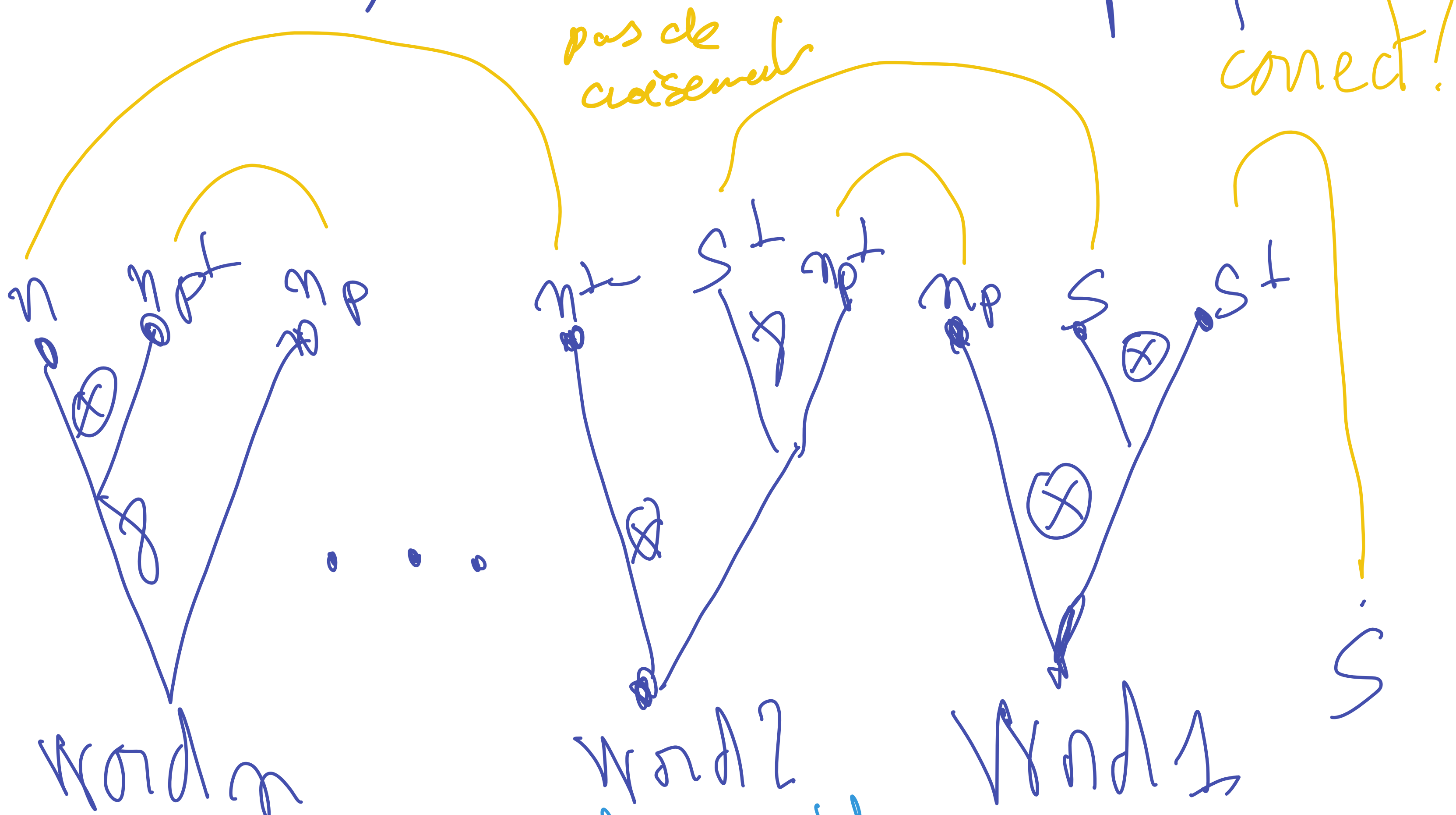
→ why?

analyses of the 7 most likely sequences of TAGs

7: in 90% of the cases the proper analysis is in the 7
to increase this 90% one needs many more sequences
THRESOLD

- machine learning
+ checking proof net correctness
- fails → exhaustive exploration

(cut elimination) $\longrightarrow$ normal proofs only



guessing where the axioms are

machine learning solution works in 70%

On going improvement

lexical semantics

I finished my textbook.

↳ read, write, print...

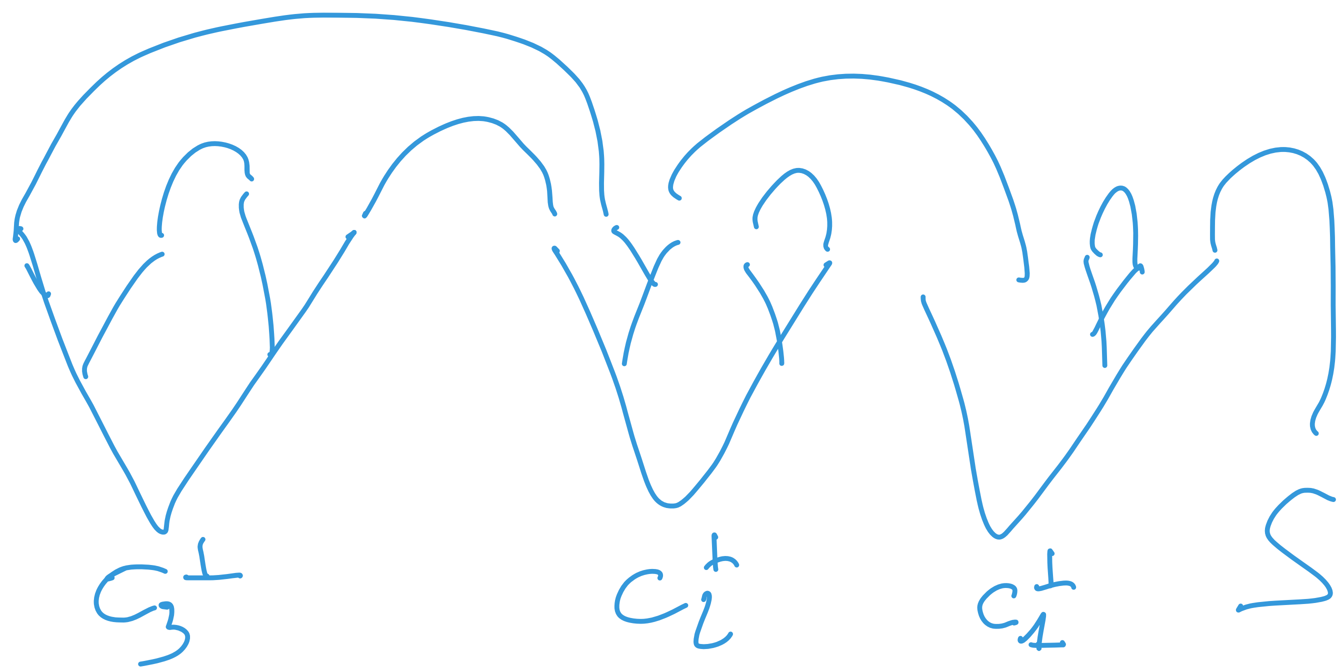
Proof nets as parse structures

- easy to construct
- easy to connect to ~~X~~ terms

+ provides additional information

$$A_1^{m_1}, \quad \dots \quad A_n^{m_n} \vdash S$$

$$\vdash \underbrace{\neg A_n^{m_n}}, \dots, \underbrace{\neg A_1^{m_1}}, S$$



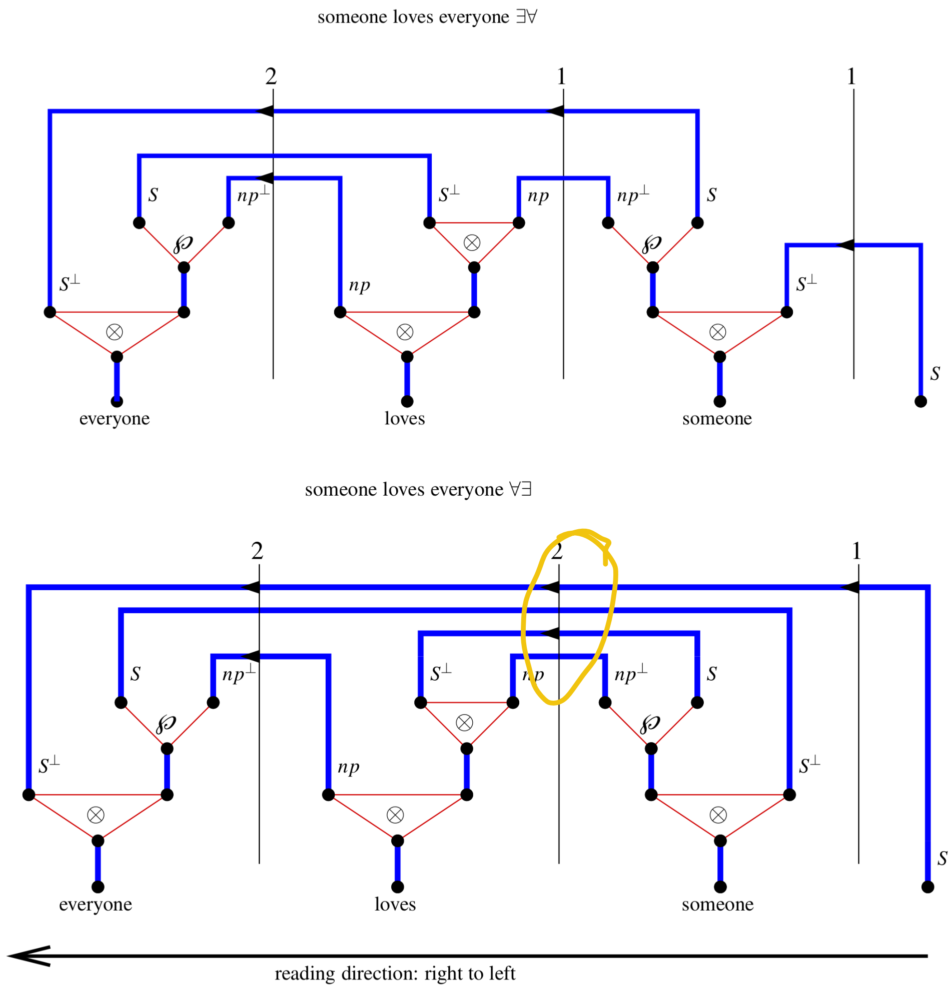


Fig. 6.6. "Someone loves everyone" with wide scope for everyone. The complexity profile — read from right to left — is $1-2-2$.

Axiom links
(especially $X \leftarrow X^{\perp}$,
missing categories)

measure

local
complexity
of human
understanding

Proof theoretical view of natural language analysis

- parse structure: proof in L of s
  intuitionistic
  non commutative
  multiplicative linear logic
- Semantic interpretation

  proof in NJ of t

proof of $A \supset B$: function mapping proofs of A to proofs of B

Bilan

grammaire de Lambek + sémantique de Montague

fonctions ! fonctions !

mathématiquement joli synt/sém parfait

mais trop restreint

MMCG extension moins jolie

mais fonctionne bien

(et efficacement avec étapes Deep Learning)

Aujourd'hui fleurissent
analyses statistiques par machine learning

→ structure syntaxique assignée ?
sens lexical par connotation ok
qu'est-ce qui est affirmé
réfuté
supposé ? portée de la négation ?

Sa souffle fort, mais ce n'est pas un ouragan.

Geach était-il un étudiant de Wittgenstein?

– En 1941, IL épousa la philosophe Elizabeth Anscombe, grâce à LAQUELLE IL entra en contact avec Ludwig Wittgenstein. BIEN_QU IL N'ait JAMAIS suivi l'enseignement académique de CE_DERNIER, cependant IL EN éprouva fortement l'influence. [Wikipedia]

Geach n'est pas un étudiant de Wittgenstein

Merci.

Des questions