Which type theory for lexical semantics?

Christian Retoré (Université de Bordeaux, LaBRI & INRIA)
Joint work with Bruno Mery and Christian Basac

16^{e} rencontres du groupe de travail *Logique, Algèbre et Calcul*

2010, Novembre 16^{th}

Contents

Part I
Lexical issues in compositional semantics

1. Typical examples of meaning slips

- Qualia
  - *A quick cigarette* (telic)
  - *A partisan article* (agentive)
- Dot Objects
  - *An interesting book* (I)
  - *A heavy book* (φ)
  - *A large city* (T)
  - *A cosmopolitan city* (P)

2. Typical examples of copredication

- Co-predications
  - *A heavy, yet interesting book*
  - *Paris is a large, cosmopolitan city*
  - ? *A fast, delicious salmon*
  - ?? *Washington is a small city of the East coast and attacked Irak*

Part II

The usual framework: Montague semantics

3. Back to the roots: Montague semantics. Types.

Simply typed lambda terms

types ::= $e \mid t \mid$ *types* $\rightarrow$ *types*

chair , *sleep* $e \rightarrow t$

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

4. Back to the roots: Montague semantics. Syntax/semantics.

(Syntactic type)*	=	Semantic type	
S^*	=	t	a sentence is a proposition
np^*	=	e	a noun phrase is an entity
n^*	=	$e \rightarrow t$	a noun is a subset of the set of entities
$(A\backslash B)^* = (B/A)^*$	=	$A \rightarrow B$	extends easily to all syntactic categories of a Categorial Grammar e.g. a Lambek CG

5. Back to the roots: Montague semantics. Logic within lambda-calculus 1/2.

Logical operations (and, or, some, all the,.....) need constants:

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

6. Back to the roots: Montague semantics. Logic within lambda-calculus 2/2.

Words in the lexicon need constants for their denotation:

likes	$\lambda x \lambda y\ (\mathrm{likes}\ y)\ x$	$x : e,\ y : e, \mathrm{likes} : e \to (e \to t)$
« likes » is a two-place predicate		
Garance	$\lambda P\ (P\ \mathrm{Garance})$	$P : e \to t,\ \mathrm{Garance} : e$
« Garance » is viewed as the properties that « Garance » holds		

7. Back to the roots: 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 its normal form corresponds to a formula, the "meaning".

8. Back to the roots: Montague semantics. Computing the semantics. 2/5

word	***semantic type*** u^* ***semantics :*** λ***-term of type*** u^* x_v *the variable or constant* x *is of type* v
some	$(e \rightarrow t) \rightarrow ((e \rightarrow t) \rightarrow t)$ $\lambda P_{e\rightarrow t}\ \lambda Q_{e\rightarrow t}\ (\exists_{(e\rightarrow t)\rightarrow t}\ (\lambda x_e(\wedge_{t\rightarrow(t\rightarrow t)}(P\ x)(Q\ x)$
statements	$e \rightarrow t$ $\lambda x_e(\mathtt{statement}_{e\rightarrow t}\ x)$
speak_about	$e \rightarrow (e \rightarrow t)$ $\lambda y_e\ \lambda x_e\ ((\mathtt{speak\_about}_{e\rightarrow(e\rightarrow t)}\ x)y)$
themselves	$(e \rightarrow (e \rightarrow t)) \rightarrow (e \rightarrow t)$ $\lambda P_{e\rightarrow(e\rightarrow t)}\ \lambda x_e\ ((P\ x)x)$

9. Back to the roots: 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) (themsleves speak_about)) of type *t*

10. Back to the roots: Montague semantics. Computing the semantics. 4/5

$$
\begin{gathered}
\Big((\lambda P_{e\to t}\ \lambda Q_{e\to t}\ (\exists_{(e\to t)\to t}\ (\lambda x_e(\wedge(P\ x)(Q\ x)))))\\
(\lambda x_e(\texttt{statement}_{e\to t}\ x))\Big)\\
\Big((\lambda P_{e\to(e\to t)}\ \lambda x_e\ ((P\ x)x))\\
(\lambda y_e\ \lambda x_e\ ((\texttt{speak\_about}_{e\to(e\to t)}\ x)y))\Big)
\end{gathered}
$$

$$\downarrow \beta$$

$$
\begin{gathered}
(\lambda Q_{e\to t}\ (\exists_{(e\to t)\to t}\ (\lambda x_e(\wedge_{t\to(t\to t)}(\texttt{statement}_{e\to t}\ x)(Q\ x)))))\\
(\lambda x_e\ ((\texttt{speak\_about}_{e\to(e\to t)}\ x)x))
\end{gathered}
$$

$$\downarrow \beta$$

$$(\exists_{(e\to t)\to t}\ (\lambda x_e(\wedge(\texttt{statement}_{e\to t}\ x)((\texttt{speak\_about}_{e\to(e\to t)}\ x)x))))$$

11. Back to the roots: Montague semantics. Computing the semantics. 5/5

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

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

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

Part III

Extending the type system

12. More general types and terms. Many sorted logic. *TY_n*

Extension to TY_n without difficulty nor suprise: e can be divided in several kind of entities.
It's a kind of flat ontology: objects, concepts, events,...

13. More general types and terms. Second order types (Girard's F).

One can also add type variables and quantification over types.

- Constants e and t, as well as any type variable α in P, are types.
- Whenever T is a type and α a type variable which may but need not occur in T, $\Lambda\alpha.\ T$ is a type.
- Whenever T_1 and T_2 are types, $T_1 \rightarrow T_2$ is also a type.

14. More general types and terms. Second order terms (Girard's F).

- A variable of type T i.e. $x : T$ or x^T is a *term*. Countably many variables of each type.

- $(f\ \tau)$ is a term of type U whenever $\tau : T$ and $f : T \to U$.

- $\lambda x^T.\ \tau$ is a term of type $T \to U$ whenever $x : T$, and $\tau : U$.

- $\tau\{U\}$ is a term of type $T[U/\alpha]$ whenever $\tau : \Lambda\alpha.\ T$, and U is a type.

- $\Lambda\alpha.\tau$ is a term of type $\Lambda\alpha.T$ whenever α is a type variable, and $\tau : T$ without any free occurrence of the type variable α.

15. More general types and terms. Second order reduction.

The reduction is defined as follows:

- $(\Lambda\alpha.\tau)\{U\}$ reduces to $\tau[U/\alpha]$ (remember that α and U are types).
- $(\lambda x.\tau)u$ reduces to $\tau[u/x]$ (usual reduction).

16. More general types and terms. A second order example.

Given two predicates $P^{\alpha \to t}$ and $Q^{\beta \to t}$
over entities of respective kinds α and β
when we have two morphisms from ξ to α and to β
we can coordinate entities of type ξ:
$\Lambda\xi \lambda x^{\xi} \lambda f^{\xi \to a} \lambda g^{\xi \to b}.(\mathit{and}\ (P\ (f\ x))(Q\ (g\ x)))$

One can even quantify over the predicates P, Q and the types α, β to which they apply:
$\Lambda\alpha \Lambda\beta \lambda P^{\alpha \to t} \lambda Q^{\beta \to t} \Lambda\xi \lambda x^{\xi} \lambda f^{\xi \to \alpha} \lambda g^{\xi \to \beta}.(\mathrm{and}\ (P\ (f\ x))(Q\ (g\ x)))$

Part IV
Integrating facets in a compositional lexicon

17. Principles of our lexicon

- Remain within realm of Montagovian compositional semantics (but no models).
- Allow both predicate and argument to contribute lexical information to the compound.
- Integrate within existing discourse models (λ-DRT).

We advocate a system based on *optional modifiers*.

18. The Types

- Montagovian composition:
  - Predicate include the typing and the order of its arguments.
- Generative Lexicon style concept hierarchy:
  - Types are different for every distinct lexical behavior
  - A kind of ontology details the specialization relations between types

Second-order typing, like Girard's F system is needed for arbitrary modifiers:

$$\Lambda\alpha\lambda x^A y^\alpha f^{\alpha\to R}.((\mathrm{read}^{A\to R\to t}\ x)\ (f\ y))$$

19. The Terms: main / standard term

- A standard λ-term attached to the main sense:
  - Used for compositional purposes
  - Comprising detailed typing information
  - Including slots for optional modifiers
  - e.g. $\Lambda\alpha\beta\lambda x^{\alpha} y^{\beta} f^{\alpha\to A} g^{\beta\to F}.((\mathrm{eat}^{A\to F\to t}\ (f\ x))\ (g\ y))$
  - e.g. Paris^{T}

20. The Terms: Optional Morphisms

- Each a one-place predicate
- Used, or not, for adaptation purposes
- Each associated with a constraint : *rigid*, $\varnothing$

$$* \left(\frac{Id^{F \rightarrow F}}{\varnothing}, \frac{f_{grind}^{Living \rightarrow F}}{rigid} \right)$$

$$* \left(\frac{Id^{T \rightarrow T}}{\varnothing}, \frac{f_{L}^{T \rightarrow L}}{\varnothing}, \frac{f_{P}^{T \rightarrow P}}{\varnothing}, \frac{f_{G}^{T \rightarrow G}}{rigid} \right)$$

21. A Complete Lexical Entry

Every lexeme is associated to an *n*-uple such as:

$$\left(\text{Paris}^T, \frac{\lambda x^T.\ x^T}{\varnothing}, \frac{\lambda x^T.(f_L^{T\to L}\ x)}{\varnothing}, \frac{\lambda x^T.(f_P^{T\to P}\ x)}{\varnothing}, \frac{\lambda x^T.(f_G^{T\to G}\ x)}{\mathit{rigid}}\right)$$

22. RIGID vs flexible use of optional morphisms

Type clash: $(\lambda x^V.\ (P^{V\to W} x))\tau^U$

$$(\lambda x^V.\ (P^{V\to W} x))\ (f^{U\to V} \tau^U)$$

f: optional term associated with either P or τ
f **applies once to the argument** and not to the several occurrences of x in the function.
A conjunction yields
$(\lambda x^V.\ (\wedge\ (P^{V\to W} x)\ (Q^{V\to W} x))\ (f^{U\to V} \tau^U)$, the argument is uniformly transformed.
Second order is not needed, the type V of the argument is known and it is always the same for every occurrence of x.

23. FLEXIBLE vs. rigid use of optional morphisms

$(\lambda x^?.\ (\cdots(P^{A\to X} x^?)\cdots(Q^{B\to Y} x^?)\cdots)\tau^U$:

type clash(es) [Montague: ? = A = B e.g. $e \to t$]

$$(\Lambda\xi.\lambda f^{\xi\to A}.\lambda g^{\xi\to B}.\ (\cdots(P^{A\to X}(f x^{\xi}))\cdots(Q^{B\to Y}(g x^{\xi}))\cdots))$$
$$\{U\}\ f^{U\to A}\ g^{U\to B}\ \tau^U$$

f, g: optional terms associated with either P or τ.
For each occurrence of x
with different A, B, ... with different f, g, ... each time.

Second order typing:
1) anticipates the yet unknown type of the argument
2) factorizes the different function types in the slots.

The types $\{U\}$ and the associated morphism f are inferred from the original formula $(\lambda x^V.\ (P^{V\to W} x))\tau^U$.

24. Standard behaviour

ϕ: physical objects

small stone

$$\overbrace{(\lambda x^{\varphi}.\ (\text{small}^{\varphi\rightarrow\varphi}\, x))}^{\text{small}}\ \overbrace{\tau^{\varphi}}^{\text{stone}}$$

$$(\text{small}\ \tau)^{\varphi}$$

25. Qualia exploitation

wondering, loving smile

$$\overbrace{(\lambda x^{P}.\ (\mathrm{and}^{t\to(t\to t)}\ (\mathrm{wondering}^{P\to t}\ x)\ (\mathrm{loving}^{P\to t}\ x)))}^{\text{wondering, loving}}\ \overbrace{\tau^{S}}^{\text{smile}}$$

$$(\lambda x^{P}.\ (\mathrm{and}^{t\to(t\to t)}\ (\mathrm{wondering}^{P\to t}\ x)\ (\mathrm{loving}^{P\to t}\ x))))(f_a^{S\to P}\tau^{S})$$

$$(\mathrm{and}\ (\mathrm{loving}\ (f_a\ \tau))\ (\mathrm{loving}\ (f_a\ \tau)))$$

26. Facets (dot-objects): incorrect copredication

Incorrect co-predication. The rigid constraint blocks the copredication e.g. $f_g^{Fs \to Fd}$ cannot be **rigidly** used in

(??) *The tuna we had yesterday was lightning fast and delicious.*

27. Facets, correct co-predication. Town example 1/3

T town L location P people
$f_p^{T \to P}$ $f_l^{T \to L}$ k^T København

København is both a seaport and a cosmopolitan capital.

28. Facets, correct co-predication. Town example 2/3

Conjunction of $cospl^{P\to t}$, $\mathrm{cap}^{T\to t}$ and $port^{L\to t}$, on k^T
If $T = P = L = e$, (Montague)
$(\lambda x^e(\mathrm{and}^{t\to(t\to t)}((\mathrm{and}^{t\to(t\to t)}\ (\mathrm{cospl}\ x)\ (\mathrm{cap}\ x))\ (\mathrm{port}\ x)))\ k.$
Here *AND* between three predicates over different kinds $P^{\alpha\to t}$, $Q^{\beta\to t}$, $R^{\beta\to t}$

$$\begin{array}{l}\Lambda\alpha\Lambda\beta\Lambda\gamma\\ \quad\lambda P^{\alpha\to t}\lambda Q^{\beta\to t}\lambda R^{\gamma\to t}\\ \quad\quad\Lambda\xi\lambda x^{\xi}\\ \quad\quad\quad\lambda f^{\xi\to\alpha}\lambda g^{\xi\to\beta}\lambda h^{\xi\to\gamma}.\\ \quad\quad\quad\quad(\mathrm{and}(\mathrm{and}\ (P\ (f\ x))(Q\ (g\ x)))(R\ (h\ x)))\end{array}$$

f, g and h convert x to **different** types.

29. Facets, correct co-predication. Town example 3/3

AND applied to P and T and L and to $cospl^{P\to t}$ and $cap^{T\to t}$ and $\mathrm{port}^{L\to t}$ yields:

$$\Lambda\xi\lambda x^{\xi}\lambda f^{\xi\to\alpha}\lambda g^{\xi\to\beta}\lambda h^{\xi\to\gamma}.$$
$$(\mathrm{and}(\mathrm{and}\ (cospl^{P\to t}\ (f_p\ x))(cap^{T\to t}\ (f_t\ x)))(\mathrm{port}^{L\to t}\ (f_l\ x)))$$

We now wish to apply this to the type T and to the transformations provided by the lexicon. No type clash with $cap^{T\to t}$, hence $id^{T\to T}$ works. For L and P we use the transformations f_p and f_l.

$$(\mathrm{and}^{t\to(t\to t)}$$
$$(\mathrm{and}^{t\to(t\to t)}$$
$$(\mathrm{cospl}(f_p\ k^T)^P)^t)(\mathrm{cap}(\mathrm{id}\ k^T)^T)^t)^t(\mathrm{port}\ (f_l\ k^T)^L)^t)^t$$

30. The calculus, summarized

- First-order λ-bindings: usual composition
- Open slots: generate all combinations of modifiers available
- As many interpretations as well-typed combinations

Paris is an populous city by the Seine river

$$((\Lambda\xi\ .\ \lambda x^{\xi} f^{\xi\rightarrow P} g^{\xi\rightarrow L}\ .(\mathrm{and}(\mathrm{populous}^{P\rightarrow t}(f\ x))(\mathrm{riverside}^{L\rightarrow t}(g\ x)))$$

$$\{T\}\ \mathrm{Paris}^{T}\ \lambda x^{T}\ (f_{P}^{T\rightarrow P}\ x)\ \lambda x^{T}\ .\ (f_{L}^{T\rightarrow L}\ x))$$

31. Logical Formulæ

- Many possible results
- Our choice: classical, higher-order predicate logic
- No modalities

$$\text{and}(\text{populous}(f_P(\text{Paris})),\ \text{riverside}(f_L(\text{Paris})))$$

Part V

Intermezzi: tricky questions

32. Counting — Situation

A shelf.

- Three copies of *Madame Bovary*.
- Two copies of *L'éducation sentimentale*.
- The collected novels of Flaubert in one volume (*L'éducation sentimentale, Madame Bovary, Bouvard et Pécuchet*)
- A volume contains *Trois contes: Un coeur simple, La légende de Saint-Julien, Salammbô*
- One copy of the two volume set called *Correspondance*.

33. Counting — Questions

- I carried down all the books to the cellar.
- Indeed, I read them all.

- How many books did you carry?
- How many books did you read?

34. Counting — Solution

Solved by projection,
count **after** the appropriate transformation,
pronouns refer to noun phrase **before** transformation.

Provided the language issue is made clear.
(*book*$\neq$*livre*)

Similar to:
Raccoons settled in the garage.
They give live births.

35. Influence of syntax

When one of the two predicates is nested within a syntactic clause, copredication can become felicitous.

* This lightning fast salmon is delicious.
?? This once lightning fast salmon is delicious.
This salmon that used to be lightning fast is delicious.
(Not a yes/no acceptability.)

Modeled by unlocking the rigidity condition.

Part VI
Critics, towards a linear alternative

36. Critics

- The classical solution with products: forces $\langle p_1(u), p_2(u)\rangle = u$ (doubtful)
- (Asher's solution with pullbacks) too tight relation type structure / morphisms (only and always canonical morphisms) and unavoidable relation to product
- (Ours) not enough relation types/morphisms (no relation at all), typing does not constrain morphims,

37. Language vs. (discourse) universe

How things are and works / Lexical description

Ambiguity: does the lexicon describe

- the world of the discourse universe (ontology)
- or a language dependent ontology:

Ma voiture est crevée. even *J'ai crevé.*
(une roue de ma voiture est crevée).

but

* *Ma voiture est bouchée. (le carburateur)*

* *Ma voiture est à plat. (la batterie)*

38. Language variation is mainly lexical

This shows there is a language dependent way for words and pronouns to access facets.

Such examples as well as cross linguistic comparisons indicate a distinction should be made.

Language acts as an idiosyncratic filter over the (discourse) universe — we can possibly model this.

Language also creates specific connections (captivus: cattivo vs. chétif, morbus: morbide vs. morbido) — more difficult to model.

39. Linear alternative

Direct representation with monoidal product $A \otimes B$ and replication !

- $A \otimes B$
  - without $\langle p_1(u), p_2(u) \rangle = u$
  - without canonical morphism(s)
  - but the type of a transformation relates to the structure of the type.
- Types of morphisms in a linear setting either:
  - <u>irreversible:</u> $A \multimap U$ since $A \not\multimap U \otimes A$
  - <u>reusable:</u> $A \rightarrow B = (!A) \multimap U$ since $(!A) \multimap U \otimes (!A)$

Part VII

Conclusion

40. Our solution and further studies

Extension of Montague semantics with type modifications: already implemented in the categorial parser Grail devloped by Richard Moot.

- Grammar extracted from regional historical corpus, Le Monde
- Semantics relations difficult to extract, mainly and written data in a small part of the lexicon.

The linear model: study of first order linear logic, in particular models. First exploring intuitionistic models, in particular sheaf models.

Longue et heureuse retraite à René!