Which type theory for lexical semantics?

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

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

2010, Novembre 16\textsuperscript{th}
Contents

I Lexical issues in compositional semantics 3

II The usual framework: Montague semantics 6

III Extending the type system 16

IV Integrating facets in a compositional lexicon 22

V Intermezzi: tricky questions 38

VI Critics, towards a linear alternative 43

VII Conclusion 48
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* \((\varphi)\)
  - *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

Simply typed lambda terms

\[
\text{types} ::= e \mid t \mid \text{types} \rightarrow \text{types}
\]

chair, sleep \(e \rightarrow t\)

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

<table>
<thead>
<tr>
<th>(Syntactic type)*</th>
<th>=</th>
<th>Semantic type</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S^*$ = $t$</td>
<td></td>
<td>a sentence is a proposition</td>
</tr>
<tr>
<td>$np^*$ = $e$</td>
<td></td>
<td>a noun phrase is an entity</td>
</tr>
<tr>
<td>$n^*$ = $e \rightarrow t$</td>
<td></td>
<td>a noun is a subset of the set of entities</td>
</tr>
<tr>
<td>$(A \backslash B)^<em>$ = $(B / A)^</em>$ = $A \rightarrow B$</td>
<td></td>
<td>extends easily to all syntactic categories of a Categorial Grammar e.g. a Lambek CG</td>
</tr>
</tbody>
</table>
5. **Back to the roots: Montague semantics.**

**Logic within lambda-calculus 1/2.**

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

<table>
<thead>
<tr>
<th>Constant</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>∃</td>
<td>(e → t) → t</td>
</tr>
<tr>
<td>∀</td>
<td>(e → t) → t</td>
</tr>
<tr>
<td>∧</td>
<td>t → (t → t)</td>
</tr>
<tr>
<td>∨</td>
<td>t → (t → t)</td>
</tr>
<tr>
<td>⊃</td>
<td>t → (t → t)</td>
</tr>
</tbody>
</table>

Words in the lexicon need constants for their denotation:

<table>
<thead>
<tr>
<th>likes</th>
<th>( \lambda x \lambda y \ (\text{likes } y) \ x )</th>
<th>( x : e, y : e, \text{likes} : e \rightarrow (e \rightarrow t) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>« likes » is a two-place predicate</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Garance</th>
<th>( \lambda P \ (P \ \text{Garance}) )</th>
<th>( P : e \rightarrow t, \text{Garance} : e )</th>
</tr>
</thead>
<tbody>
<tr>
<td>« Garance » is viewed as</td>
<td></td>
<td></td>
</tr>
<tr>
<td>the properties that « Garance » holds</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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 $\lambda$-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

<table>
<thead>
<tr>
<th>word</th>
<th>semantic type $u^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>semantics</td>
<td>$\lambda$-term of type $u^*$</td>
</tr>
<tr>
<td>$x_v$ the variable or constant $x$ is of type $v$</td>
<td></td>
</tr>
</tbody>
</table>

| 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      | $e \rightarrow t$ |
|                 | $\lambda x_{e}(\text{statement}_{e\rightarrow t} x)$ |

| speak_about     | $e \rightarrow (e \rightarrow t)$ |
|                 | $\lambda y_{e} \lambda x_{e} ((\text{spea}_k\text{t}_a\text{bou}_k\text{t}_{e\rightarrow(e\rightarrow t)} x)y)$ |

| themselves      | $(e \rightarrow (e \rightarrow t)) \rightarrow (e \rightarrow t)$ |
|                 | $\lambda P_{e\rightarrow(e\rightarrow t)} \lambda x_{e} ((P x) x)$ |
Computing the semantics. 3/5

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

$$((\text{some statements}) \ (\text{themselves speak\_about})) \text{ of type } t$$
Computing the semantics. 4/5

\[
((\lambda P_{e \rightarrow t} \lambda Q_{e \rightarrow t} (\exists (e \rightarrow t) \rightarrow t (\lambda x_e (\wedge (P x)(Q x))))))
\]
\[
(\lambda x_e (\text{statement}_{e \rightarrow t} x)))
\]
\[
((\lambda P_{e \rightarrow (e \rightarrow t)} \lambda x_e ((P x)x))
\]
\[
(\lambda y_{e} \lambda x_e ((\text{speak}_{e \rightarrow (e \rightarrow t)} x)y)))
\]
\[
\downarrow \beta
\]
\[
(\lambda Q_{e \rightarrow t} (\exists (e \rightarrow t) \rightarrow t (\lambda x_e (\wedge (t \rightarrow (t \rightarrow t})(\text{statement}_{e \rightarrow t} x)(Q x))))))
\]
\[
(\lambda x_e ((\text{speak}_{e \rightarrow (e \rightarrow t)} x)x))
\]
\[
\downarrow \beta
\]
\[
(\exists (e \rightarrow t) \rightarrow t (\lambda x_e (\wedge (\text{statement}_{e \rightarrow t} x)(((\text{speak}_{e \rightarrow (e \rightarrow t)} x)x))))
\]
Computing the semantics. 5/5

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

$$\exists x : e \ (\text{statement}(x) \land \text{speak}_{-}\text{about}(x, x))$$

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. $T \mathcal{Y}_n$

Extension to $T \mathcal{Y}_n$ without difficulty nor surprise: $e$ 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 \( \alpha \) in \( P \), are types.
- Whenever \( T \) is a type and \( \alpha \) 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 $f : T \rightarrow U$.

• $\lambda x^T. \tau$ is a term of type $T \rightarrow U$ 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 $\alpha$ is a type variable, and $\tau : T$ without any free occurrence of the type variable $\alpha$. 

The reduction is defined as follows:

- $(\Lambda \alpha \cdot \tau)\{U\}$ reduces to $\tau[U/\alpha]$ (remember that $\alpha$ and $U$ are types).

- $(\lambda x \cdot \tau)u$ reduces to $\tau[u/x]$ (usual reduction).

Given two predicates $P^{\alpha \to t}$ and $Q^{\beta \to t}$ over entities of respective kinds $\alpha$ and $\beta$ when we have two morphisms from $\xi$ to $\alpha$ and to $\beta$ we can coordinate entities of type $\xi$:

$$\Lambda_{\xi} \lambda x^{\xi} \lambda f^{\xi \to a} \lambda g^{\xi \to b}.(\text{and} (P (f \ x))(Q (g \ x)))$$

One can even quantify over the predicates $P$, $Q$ and the types $\alpha$, $\beta$ to which they apply:

$$\Lambda_{\alpha} \Lambda_{\beta} \lambda P^{\alpha \to t} \lambda Q^{\beta \to t} \Lambda_{\xi} \lambda x^{\xi} \lambda f^{\xi \to a} \lambda g^{\xi \to b}.(\text{and} (P (f \ x))(Q (g \ x)))$$
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 ($\lambda$-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:

\[ \Lambda x^A y^\alpha \rightarrow^R \alpha (\text{read}^A \rightarrow^R \rightarrow^t x) (f y) \]
19. The Terms: main / standard term

- A standard $\lambda$-term attached to the main sense:
  - Used for compositional purposes
  - Comprising detailed typing information
  - Including slots for optional modifiers
  - e.g.
    \[ \Lambda_{\alpha\beta} \lambda x^\alpha y^\beta f^{\alpha\rightarrow A} g^{\beta\rightarrow F}.((\text{eat}^{\alpha\rightarrow F\rightarrow t} (f \ x)) \ (g \ y)) \]
  - 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: \( \text{rigid}, \emptyset \)

\[
\begin{align*}
&* \left( \begin{array}{c}
\text{Id}_{F \to F} \\
\emptyset \\
\text{Living} \\
\text{grind} \\
\text{rigid}
\end{array}, \begin{array}{c}
\text{Id}_{T \to T} \\
\emptyset \\
\text{Living}_{L \to P} \\
\text{grind} \\
\text{rigid}
\end{array}, \begin{array}{c}
\text{Id}_{P \to P} \\
\emptyset \\
\text{grind} \\
\text{rigid}
\end{array}, \begin{array}{c}
\text{Id}_{G \to G} \\
\text{rigid}
\end{array} \right)
\end{align*}
\]
21. A Complete Lexical Entry

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

\[
\left( \text{Paris}^T, \frac{\lambda x^T. x^T}{\emptyset}, \frac{\lambda x^T.(f_{L}^{T \to L} x)}{\emptyset}, \frac{\lambda x^T.(f_{P}^{T \to P} x)}{\emptyset}, \frac{\lambda x^T.(f_{G}^{T \to G} x)}{\text{rigid}} \right)
\]
22. RIGID vs flexible use of optional morphisms

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

\[
(\lambda x^V. (P^{V\rightarrow W} x)) (f^{U\rightarrow V}_{\tau^U})
\]

\(f\): optional term associated with either \(P\) or \(\tau\)

\(f\) 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\rightarrow W} x) (Q^{V\rightarrow W} x)) (f^{U\rightarrow 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**

\[(\lambda x^? \cdot (\ldots (P^A \to^X x^?) \ldots (Q^B \to^Y x^?) \ldots)_{\tau U}: \tau U, \ U, \ fU \to A, \ gU \to B)\]

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

\[(\Lambda \xi. \lambda f^\xi \to^A \lambda g^\xi \to^B \cdot (\ldots (P^A \to^X (fx^\xi)) \ldots (Q^B \to^Y (gx^\xi)) \ldots))\]

\[\{U\} \ fU \to A \ gU \to B \ \tau U\]

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 \((\lambda x^V \cdot (P^V \to^W x))_{\tau U}\).
24. Standard behaviour

$\phi$: physical objects

\[
\text{small stone}
\]

\[
(\lambda x^\varphi. (\text{small}^\varphi x))^{\tau^\varphi}
\]

\[
(\text{small}^{\tau})^\varphi
\]
25. Qualia exploitation

wondering, loving smile

\[
\begin{align*}
&\text{wondering, loving} \\
&\quad\left(\lambda x^P. (\text{and}^t \to (t \to t) (\text{wondering}^{P \to t} x) (\text{loving}^{P \to t} x))\right) \\
&\quad\left(\lambda x^P. (\text{and}^t \to (t \to t) (\text{wondering}^{P \to t} x) (\text{loving}^{P \to t} x))\right) (f^S_{a \to P \to S}) \\
&\quad(\text{and} (\text{loving} (f^T_{a \to T})) (\text{loving} (f^T_{a \to T})))
\end{align*}
\]
26. Facets (dot-objects): incorrect copredication

Incorrect co-predication. The rigid constraint blocks the copredication e.g. $f^{\text{Fs} \rightarrow \text{Fd}}_g$ cannot be \textbf{rigidly} used in

(??) The tuna we had yesterday was lightning fast and delicious.
27. Facets, correct co-predication. Town example 1/3

\[ T \text{ town} \quad L \text{ location} \quad P \text{ people} \]
\[ f_p^{T \to P} \quad f_l^{T \to L} \quad k^T \text{ København} \]

København is both a seaport and a cosmopolitan capital.
28. Facets, correct co-predication. Town example 2/3

Conjunction of $\text{cospl}^{P \to t}$, $\text{cap}^{T \to t}$ and $\text{port}^{L \to t}$, on $k_T$

If $T = P = L = e$, (Montague)

$$(\lambda x^e (\text{and}^{t \to (t \to t)} ((\text{and}^{t \to (t \to t)} (\text{cospl} x) (\text{cap} x)) (\text{port} x)))) \lambda t \to (t \to t).$$

Here AND between three predicates over different kinds $P^{\alpha \to t}$, $Q^{\beta \to t}$, $R^{\gamma \to t}$

$$\Lambda \alpha \Lambda \beta \Lambda \gamma$$

$$\lambda P^{\alpha \to t} \lambda Q^{\beta \to t} \lambda R^{\gamma \to t}$$

$$\Lambda \xi \lambda x^{\xi}$$

$$\lambda f^{\xi \to \alpha} \lambda g^{\xi \to \beta} \lambda h^{\xi \to \gamma}.$$ 

$$(\text{and}(\text{and}(P (f x)) (Q (g x))) (R (h x)))$$

$f, g$ 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 $\text{cospl}^P \rightarrow t$ and $\text{cap}^T \rightarrow t$ and $\text{port}^L \rightarrow t$ yields:

$$\Lambda \xi \lambda x^\xi \lambda f^\xi \rightarrow \alpha \lambda g^\xi \rightarrow \beta \lambda h^\xi \rightarrow \gamma.$$

$$(\text{and}(\text{and} (\text{cospl}^P \rightarrow t (f_P x))(\text{cap}^T \rightarrow t (f_t x))(\text{port}^L \rightarrow t (f_l x))))$$

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

$$(\text{and}^t \rightarrow (t \rightarrow t)$$

$$(\text{and}^t \rightarrow (t \rightarrow t)$$

$$(\text{cospl}(f_Pk^T)^P t)(\text{cap}(id k^T)^T t)^t(\text{port} (f_l k^T)^L t)^t$$
30. The calculus, summarized

- First-order $\lambda$-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*

$$(((\Lambda \xi . \lambda x^\xi f^{\xi \rightarrow P} g^{\xi \rightarrow L} \cdot \text{and}(\text{populous}^{P \rightarrow t}(f \ x))(\text{riverside}^{L \rightarrow t}(g \ x)))) \{ T \}) \text{ Paris}^T \lambda x^T (f_P^{T \rightarrow P} x) \lambda x^T \cdot (f_L^{T \rightarrow L} x))$$
31. Logical Formulae

- 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
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* ≠ *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:
  - irreversible: $A \leadsto U$ since $A \not\leadsto U \otimes A$
  - reusable: $A \rightarrow B = (!A) \leadsto U$ since $(!A) \leadsto U \otimes (!A)$
Part VII

Conclusion
40. Our solution and further studies

Extension of Montague semantics with type modifications: already implemented in the categorial parser Grail developed 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!