From Montague Semantics to MTT-Semantics: A Meaningful Comparison

Stergios Chatzikyriakidis and Zhaohui Luo
(ESSLLI 2019: Lecture 2 by ZL)
MTT-semantics: some key features

❖ MTT-semantic features in comparison with Montague
  ❖ Overview, some detailed further in later lectures
  ❖ Only focussing on several, others omitted
❖ Features of MTT-semantics (& differences with MG)
  ❖ Rich type structure
  ❖ Powerful tools for semantic constructions
  ❖ Both model-theoretic and proof-theoretic
  ❖ Others (if time permits: eg, judgemental interpretation and identity criteria)
I. Rich type structures & CNs-as-types

- In MTTs, many types with rich structures
  - Dependent types (Π-types, Σ-types, ...)
  - Inductive types (Nat, Fin(n), ...)
  - Universes – types of types (logical, linguistic, ...)

In contrast, simple type theory has only e, t, α → β.

- In linguistic semantics, this allows CNs-as-types.
  - For example, adjectival modification (see SC’s lecture)

<table>
<thead>
<tr>
<th>Classification</th>
<th>Inference</th>
<th>Example</th>
<th>MTT-types/mechanisms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intersective</td>
<td>Adj[N] ⇒ N &amp; Adj</td>
<td>handsome man</td>
<td>Σ-types</td>
</tr>
<tr>
<td>Subsective</td>
<td>Adj[N] ⇒ N</td>
<td>large mouse</td>
<td>Π-polymorphism</td>
</tr>
<tr>
<td>Privative</td>
<td>Adj[N] ⇒ ¬N</td>
<td>fake gun</td>
<td>disjoint union types</td>
</tr>
<tr>
<td>Non-committal</td>
<td>Adj[N] ⇒ ?</td>
<td>alleged criminal</td>
<td>modal collection</td>
</tr>
</tbody>
</table>
Several advantages of CNs-as-types (not predicates)

- Selection restriction by typing
  - Capturing category errors
  - MG cannot do this due to CNs-as-predicates.

- Compatibility with subtyping
  - Subtypes or "subsorts" (cf, Asher, Partee): Phy, Event, Containers, ...
  - MG has been problematic in this because of CNs-as-predicates.

- Proper treatment of copredication
  - Dot-types in MTTs (see SC’s latter lecture)
  - Problematic if CNs-as-predicates
I.1. Selection Restriction

❖ (*) The table talks.
  ❖ Is (*) meaningful? (a category error)

❖ In MG, yes: (*) has a truth value
  ❖ talk(the table) is false in the intended model.

❖ In MTT-semantics, no: (*) is not meaningful.
  ❖ “the table” is of type Table, not of type Human and, hence, talk(the table) is ill-typed as talk requires that its argument be of type Human.
  ❖ In MTT-semantics, meaningfulness = well-typedness
I.2. Compatibility with subtyping

- Researchers considered various subtypes/subsorts:
  - Phy, Info ≤ e (Asher 2011 on copredication)
  - Basket ≤ Container, ... (Partee-Borschev 2014, on adjectival modification)
  - Events ... (c.f., ZL’s lecture on Friday)

- Such are incompatible with CNs-as-predicates, although they are OK with CNs-as-types.
  - Let’s consider an example – copredication.
  - I’m only using this as an example – see SC’s latter Lecture 4.
Copredication

- Copredication (Asher, Pustejovsky, ...)
  - John picked up and mastered the book.
  - The lunch was delicious but took forever.
  - The newspaper you are reading is being sued by Mia.
  - ... ...

- How to deal with this in formal semantics
  - Dot-objects (eg, Asher 2011, in the Montagovian setting)
  - It has a problem: subtyping and CNs-as-predicates strategy do not fit with reach other ...
Subtyping problem in the Montagovian setting

❖ Problematic example (in Montague with CNs-as-pred)

❖ heavy : Phy→t and book : Phy•Info→t
❖ heavy book = \( \lambda x:Phy. \) book\( (x) \land \) heavy\( (x) \) ?
❖ In order for this, we’d need Phy ≤ Phy•Info (\#)
  But, intuitively, this is not the case (the opposite should be)!
❖ A higher type of heavy : (Phy→t)→(Phy→t) would not help.

❖ In MTT-semantics, because CNs are interpreted as types, things work as intended.

❖ heavy : Phy→Prop and Book ≤ Phy•Info ≤ Phy
❖ So, heavy\( (b) \) : Prop is well-typed, for b : Book.
In MTT-semantics, CNs are types – we have:

“John picked up and mastered the book.”

\[ \text{book} \leq \text{PHY} \bullet \text{INFO} \]

\[ \text{pick up} : \text{Human} \rightarrow \text{PHY} \rightarrow \text{Prop} \]

\[ \leq \text{Human} \rightarrow \text{PHY} \bullet \text{INFO} \rightarrow \text{Prop} \]

\[ \leq \text{Human} \rightarrow [\text{book}] \rightarrow \text{Prop} \]

\[ \text{master} : \text{Human} \rightarrow \text{INFO} \rightarrow \text{Prop} \]

\[ \leq \text{Human} \rightarrow \text{PHY} \bullet \text{INFO} \rightarrow \text{Prop} \]

\[ \leq \text{Human} \rightarrow [\text{book}] \rightarrow \text{Prop} \]

Hence, both have the same type (in \text{LType} – cf, SC’s Lect 1.2) and therefore can be coordinated by “and” to form “picked up and mastered” in the above sentence.

Remark: CNs as types in MTT-semantics – so things work.
II. MTT-tools for semantic constructions

❖ Rich typing ➔ powerful tools
❖ Examples:
  ❖ $\Pi$-polymorphism via universes
  ❖ Overloading by coercive subtyping
  ❖ $\Pi/\Sigma$-organisation (omitted here)
  ❖ ... ...
II.1. Sense selection via overloading

- **Sense enumeration** (cf, Pustejovsky 1995 and others)
  - Homonymy
  - Automated selection
  - Existing treatments (eg, Asher et al via disjoint union types)
- For example,
  1. John runs quickly.
  2. John runs a bank.

  with homonymous meanings
  1. \([\text{run}]_1 : \text{Human} \rightarrow \text{Prop}\)
  2. \([\text{run}]_2 : \text{Human} \rightarrow \text{Institution} \rightarrow \text{Prop}\)

  “run” is **overloaded** – how to disambiguate?
Overloading via coercive subtyping

❖ Overloading can be represented by coercions

Eg


❖ Homonymous meanings can be represented so that automated selection can be done according to typing.

❖ Remark: This could not be done if CNs-as-predicates.
II.2. Π-polymorphism – an example mechanism

- Π-polymorphism offers several applications.
  - They are not available in MG where CNs are interpreted as predicates.
- Π-types: Informally (borrowing set-theoretical notations):
  \[ \Pi x : A. B[x] = \{ f \mid \text{for any } a : A, f(a) : B[a] \} \]
  These \( f \)'s are dependent functions.
- Example
  - \( \Pi x : \text{Human}. \text{Child}(x) \), type of functions mapping \( h \) to \( \text{Child}(h) \), the type of children of \( h \) (may be an empty type).
- Notational conventions:
  - \( A \rightarrow B \) stands for \( \Pi x : A. B(x) \) when \( x \notin \text{FV}(B) \).
  - In other words, \( A \rightarrow B \) are just special cases of Π-types.
  - So, a type theory with Π-types and Prop contains simple type theory.
Π-polymorphism – a first informal look

❖ How to model predicate-modifying adverbs (eg, quickly)?
  ❖ Informally, it can take a verb and return a verb.
  ❖ Montague: quickly : (e→t)→(e→t)
    quickly(run) : e→t
❖ MTT-semantics: quickly : (A_q→Prop)→(A_q→Prop),
  where A_q is the domain/type for quickly.
  ❖ What about other verbs? (eg, A_talk=Human, ...)
  ❖ Can we do it generically – one type of all adverbs?
❖ Π-polymorphism: quickly : ΠA:CN. (A→Prop)→(A→Prop)
❖ Question: What is CN?
  Answer: CN is a universe of types – next slide.
Universes – types of types

❖ Universe of types
  ❖ Martin-Löf introduced the notion of universe (1973, 1984)
  ❖ A universe is a type of types (Note: the collection Type of all types is not a type itself – otherwise, logical paradox.)

❖ Examples
  ❖ Math: needing a universe to define type-valued functions
    ❖ $f(n) = \mathbb{N} \times \ldots \times \mathbb{N}$ (n times)
  ❖ MTT-semantics: for example,
    ❖ $\text{CN}$ is the universe of types that are (interpretations of) CNs. We have:
      Human : $\text{CN}$, Book : $\text{CN}$, $\Sigma(\text{Man}, \text{handsome}) : \text{CN}$, ... ...
    ❖ We can then have: quickly : $\prod A : \text{CN.} (A \rightarrow \text{Prop}) \rightarrow (A \rightarrow \text{Prop})$
Modelling subsective adjectives

- Examples: large, skilful, ...
- Nature of such adjectives
  - Their meanings are dependent on the nouns they modify.
  - “a large mouse” is not a large animal (“large” in “a large mouse” is only large for mice, not for other animals/entities.)
- This leads to proposal of using $\Pi$-polymorphism:
  - large : $\Pi A : \text{CN.} \ (A \rightarrow \text{Prop})$
    - CN – type universe of all (interpretations of) CNs
  - large(Mouse) : Mouse $\rightarrow$ Prop
  - $[\text{large mouse}] = \sum x : \text{Mouse}. \ \text{large(Mouse)}(x)$
Another example – type of quantifiers

❖ Generalised quantifiers
  ❖ Examples: some, three, a/an, all, ...
  ❖ In sentences like: “Some students work hard.”

❖ With $\Pi$-polymorphism, the type of binary quantifiers is (Lungu 2014):
  $$\Pi A : CN \rightarrow (A \rightarrow \text{Prop}) \rightarrow \text{Prop}$$

For $Q$ of the above type, $N : CN$ and $V : N \rightarrow \text{Prop}$
  $$Q(N,V) : \text{Prop}$$

E.g., Some(Student, work_hard) : Prop
**CNs-as-predicates in MTTs?**

- What if using an MTT but CNs-as-predicates?
  - In an MTT, one could still formally follow Montague:
    - use a single type $e$ of all entities, and
    - use predicates of type $e \rightarrow t$ to interpret CNs.

- First, this seems unnecessary, at least.
  - Why doesn’t one just use simple type theory STT?
  - STT is a simpler “subsystem” – why much bigger system?

- Secondly, most (if not all) of the advantages would be lost ...
III. MTT-sem is both model/proof-theoretic

❖ Model-theoretic semantics (traditional)
  ❖ Meaning as denotation (Tarski, ...)
  ❖ Montague: NL → (simple TT) → set theory

❖ Proof-theoretic semantics
  ❖ Meaning as inferential use (proof/consequence)
  ❖ Gentzen, Prawitz, ..., Martin-Löf
  ❖ e.g., Martin-Löf’s meaning theory

❖ MTT-semantics
  ❖ Both model-theoretic and proof-theoretic – in what sense?
  ❖ What does this imply?
**Formal semantics in Modern Type Theories (MTT-semantics) is both model-theoretic and proof-theoretic.**

- **NL → MTT** (representational, model-theoretic)
  - MTT as meaning-carrying language with its types representing collections (or "sets") and signatures representing situations
- **MTT → meaning theory** (inferential roles, proof-theoretic)
  - MTT-judgements, which are semantic representations, can be understood proof-theoretically by means of their inferential roles
- **Z. Luo. Formal Semantics in Modern Type Theories: Is It Model-theoretic, Proof-theoretic, or Both? Invited talk at LACL14.**
MTT-semantics being model-theoretic

❖ MTTs offer powerful representations.
❖ Rich type structure
  ❖ Collections represented by types
  ❖ Eg, CNs and their adjective modifications (see earlier slides)
  ❖ Wide coverage – a major advantage of model-theoretic sem
❖ Useful contextual mechanisms – signatures
  ❖ Various phenomena in linguistic semantics
    (eg, coercion & infinity)
  ❖ Situations (incomplete world) represented by signatures
    (next slide)
MTT-semantics being model-theoretic (cont\textsuperscript{ed})

- Signatures $\Sigma$ as in (cf, Edin LF [Harper et al 1987])
  \[ \Gamma \vdash \Sigma a : A \]
  with $\Sigma = c_1:A_1, \ldots, c_n:A_n$

- New forms besides $c:A$ [Luo LACL14]
  \[ \ldots, c:A, \ldots, A \preceq c B, \ldots, c \sim a : A, \ldots \]
  - Subtyping entries (cf, Lungu’s PhD thesis 2018)
  - Manifest entries (can be emulated by coercive subtyping)

- **Theorem (conservativity)**
  
  *The extension with new signature entries preserves the meta-theoretic properties for coherent signatures.*
MTT-semantics being proof-theoretic

- MTTs are representational with proof-theoretic sem
  - Not available before – cf, use theory of meaning
- MTT-based proof technology
  - Reasoning based on MTT-semantics can be carried out in proof assistants like Coq:
    - pretty straightforward but nice application of proof technology to NL reasoning (not-so-straightforward in the past …)
  - Some Coq codes can be found in: (example next slide)
Coq code (homonymy by overloading)

```
Definition CN := Set.
Parameters Bank Institution Human Man : CN.
Parameter John : Man.

(* unit type for "run" *)
Inductive Onerun : Set := run.
Definition T1 := Human -> Prop.
Definition T2 := Human -> Institution -> Prop.
Parameter run1 : T1.
Parameter run2 : T2.

(* John runs quickly *)
Parameter quickly : forall (A:CN), (A -> Prop) -> (A -> Prop).
Definition john_runs_quickly := quickly (run:T1) John.
(* John runs a bank *)
Definition john_runs_a_bank := exists b:Bank, (run:T2) John b.
```
Why important?

- **Model-theoretic – powerful semantic tools**
  - Much richer typing mechanisms for formal semantics
  - Powerful contextual mechanism to model situations

- **Proof-theoretic – practical reasoning on computers**
  - Existing technology: proof assistants (Coq, Agda, Lego/Plastic, NuPRL)
  - Applications to NL reasoning

- **Leading to both of**
  - Wide-range modelling as in model-theoretic semantics
  - Effective inference based on proof-theoretic semantics

*Remark: MTT-semantics offers a new perspective – new possibility not available before!*
IV. Several Further Features of MTTs

❖ Other features/topics:
  ❖ Judgemental interpretations (Xue, Luo & Chatzikyriakidis 18)
  ❖ Identity criteria of CNs (Luo 2012, Chatzikyriakidis & Luo 2018 – see SC’s latter lecture)
  ❖ Proof irrelevance (Luo 2019)

❖ First, introducing the last one (and, the others if time permits)
Proof Irrelevance

❖ Example to show:
   ❖ Potential problem introduced by proof terms in MTTs (and how to solve it by proof irrelevance)
   ❖ From another angle, MTTs are very powerful for semantics.

❖ Proof irrelevance
   ❖ Any two proofs of the same proposition are the same.
   ❖ To have adequate MTT-semantics, proof irrelevance needs to be enforced in the underlying type theory.
   ❖ Eg, in impredicative TTs like UTT, we can have

\[
\Gamma \vdash P : Prop \quad \Gamma \vdash p : P \quad \Gamma \vdash q : P
\]

\[
\Gamma \vdash p = q : P
\]
Examples in NL semantics

❖ Identity criteria for CNs [Luo 12, Chatzikyriakidis & Luo 18]
  ❖ A handsome man is a pair (m,p) of type $\Sigma$(Man, handsome).
  ❖ Two handsome men are the same iff they are the same man
    $\Rightarrow$ proof irrelevance (eg, proofs of handsome(m) are the same).

❖ Counting (the same problem as above)
  ❖ Most students who passed some exams are happy.
    ❖ Most $z : [\Sigma x:Student \Sigma y:Exam.\text{pass}(x,y)].\ \text{happy}(\pi_1(z))$
    ❖ Incorrect counting that takes proofs of $\Sigma y:Exam.\text{pass}(x,y)$ into account
  ❖ I believe proof irrelevance provides a clean/easier solution.
    ❖ Most $z : [\Sigma x:Student \exists y:Exam.\text{pass}(x,y)].\ \text{happy}(\pi_1(z))$
    ❖ Correct counting by proof irrelevance (for the $\exists$-proposition)
Counting and Anaphora

❖ A problem when both are involved.
  ❖ Thanks to Justyna Grudziński for bringing this example to my attention.

❖ Most farmers who own a donkey beat it.
  ❖ (♯) Most $z : [\Sigma x : F \Sigma y : D. \text{own}(x, y)]. \text{beat}(\pi_1(z), \pi_1(\pi_2(z)))$
  ❖ Incorrect counting as proofs in $\Sigma$ are taken into account.
  ❖ Note that, if you use traditional $\exists$ for both $\Sigma$ to get correct counting, anaphora are problems (untyped – $\pi_i$ don’t exist)!

❖ A problem not solved satisfactorily before
  ❖ Sundholm (1989) realised it, but only proposed an ad hoc solution.
  ❖ Tanaka (2015) studied a similar solution (ad hoc & complicated).
A solution in UTT (or MLTT$_h$), using both $\Sigma$ and $\exists$:

Most $z : [\Sigma x : F \exists y : D. \text{own}(x, y)]$.

$\forall z' : [\Sigma y' : D. \text{own}(\pi_1(z), y')]. \text{beat}(\pi_1(z), \pi_1(z'))$

Correct counting because of proof irrelevance (for $\exists$-prop).

Correct anaphoric reference because of $\Sigma$.

Details in my LACompLing18 paper (in press)

- Title: “Proof Irrelevance in Type-Theoretical Semantics”
- First (?) to do this in a “standard” logical system, rather than non-standard ones such as the non-monotonic Dynamic Predicate Logic.
Judgemental Interpretation

- Judgements v.s. propositions
- Example
  - John is a student.
  - $j : \text{Student}$ (Here, Student is a type, not a predicate.)
- What about
  - John is not a student.
  - If John is a student, he is happy.
- The following are wrong:
  - $(\#) \neg (j : \text{Student})$ -- illegitimate
  - $(\#) (j : \text{Student})$ is not the case. -- a meta-level assertion
So, we introduce $\text{IS}(\text{Student}, x) : \text{Prop}$

- $\neg \text{IS}(\text{Student}, j)$
- $\text{IS}(\text{Student}, j) \Rightarrow \text{happy}(j)$
- These are well-typed propositions.

$\text{IS}_A(B, a)$

- Introduced axiomatically.
- Justified by heterogeneous equality in type theory.