Universes in MTT-semantics

Zhaohui Luo
Royal Holloway
University of London

This talk

- Brief introduction to
  - MTT-semantics (Formal Semantics in Modern Type Theories)
  - Universes and Π-polymorphism in type theory
- Linguistic universes
  - CN – universe of CNs
  - LType – universe for coordination
- Logical universes (and proof irrelevance for MTT-sem)
  - Prop – universe in UTT of all logical propositions
  - $PROP_U$ – "universe" of small/mere propositions in HoTT's h-logic
    - MLTT/PaT-logic – inadequate (cannot have proof irrelevance)
    - $MLTT_h$, MLTT extended with h-logic, is adequate for MTT-sem (like UTT).

I. MTT-semantics

- Montague Semantics
  - R. Montague (1930–1971) & Church's simple TT
  - Dominating in linguistic semantics since 1970s
  - Set-theoretic, using simple type theory as intermediate
  - Types ("single-sorted"): e, t, e→t, …


- MTT-semantics: formal semantics in modern type theories
  - Examples of MTTs:
    - Martin-Löf's TT: predicative (adequate for MTT-sem? Later.)
    - UTT (Luo 1994) & pCIC (Coq): impredicative (MTT-sem so far)
  - Ranta (1994): formal semantics in Martin-Löf's type theory
  - Recent development on MTT-semantics

    ➔ full-scale alternative to Montague semantics

- Recent development on rich typing in NL semantics
  - Asher, Bekki, Cooper, Grudzińska, Retoré, …
    - S. Chatzikyriakidis and Z. Luo (eds.) Modern Perspectives in Type Theoretical Sem. Springer, 2017. (Collection on rich typing & …)
  - MTT-semantics is one of these developments.
    - Z. Luo. Formal Semantics in Modern Type Theories with Coercive Subtyping. Linguistics and Philosophy, 35(6). 2012.
    - S. Chatzikyriakidis and Z. Luo. Formal Semantics in Modern Type Theories. Wiley/ISTE. (Monograph on MTT-semantics, to appear)
    - S. Chatzikyriakidis and Z. Luo. From Montague to MTTs. ESSLLI 2019.
- Advantages of MTT-semantics, including
  - Both model-theoretic & proof-theoretic – new perspective not available before.

MTT-semantics: basic categories

Category	Semantic Type
S	Prop (the type of all propositions)
CNs (book, man, …)	types (each common noun is interpreted as a type)
IV	$A \rightarrow Prop$ (A is the "meaningful domain" of a verb)
Adj	$A \rightarrow Prop$ (A is the "meaningful domain" of an adjective)
Adv	$\Pi A{:}CN.(A \rightarrow Prop) \rightarrow (A \rightarrow Prop)$ (polymorphic on CNs)

In MTT-semantics, CNs are types rather than predicates:

- "man" is interpreted as a type Man : Type.
- Man could be a structured type (say, Σ(Human,male))
- A man talked.
- $\exists m{:}Man.talk(m)$: Prop, where talk : Human→Prop and Man≤Human (subtyping – crucial for MTT-semantics; see later.)

Modelling Adjective Modification: Case Study

[Chatzikyriakidis & Luo: FG13, JoLLI17]

Classical classification	**example**	**Characterisation of** Adj(N)	**MTT-semantics**
intersective	handsome man	N & Adj	$\sum x$:Man.handsome(x)
subsective	large mouse	N (Adj depends on N)	large : ΠA:CN. A→Prop large(Mouse) : Mouse→Prop
privative	fake gun	$\neg$N	$G = G_R + G_F$ with $G_R \leq_{inl} G$, $G_F \leq_{inr} G$
non-committal	alleged criminal	nothing implied	$\exists$h:Human. $H_{h,A}(\ldots)$

- $H_{h,A}(\ldots)$ expresses, eg, "h alleges …", for various non-committal adjectives A; it uses the Leibniz equality $=_{Prop}$. [Luo 2018] (*)
- cf, work on hyperintensionality (Cresswell, Lappin, Pollard, …)

Note on Subtyping in MTT-semantics

- Simple example

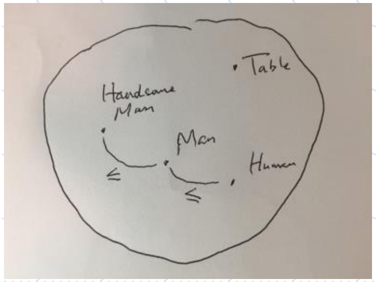
  A human talks. Paul is a handsome man.

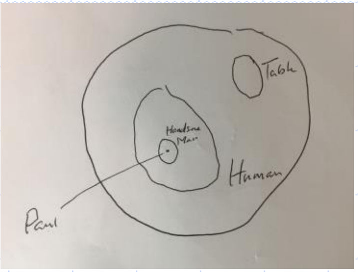
  Does Paul talk?

  Semantically, can we type talk(p)?

  (talk : Human$\rightarrow$Prop & p : Σ(Man,handsome))

  Yes, because p : Σ(Man,handsome) $\leq$ Man $\leq$ Human.





- Subtyping is crucial for MTT-semantics
  - Coercive subtyping [Luo 1999, Luo, Soloviev & Xue 2012] is adequate for MTTs and we use it in MTT-semantics.
  - Note: Traditional subsumptive subtyping is inadequate for MTTs (eg, canonicity fails with subsumption.)

Advanced features in MTT-semantics: examples

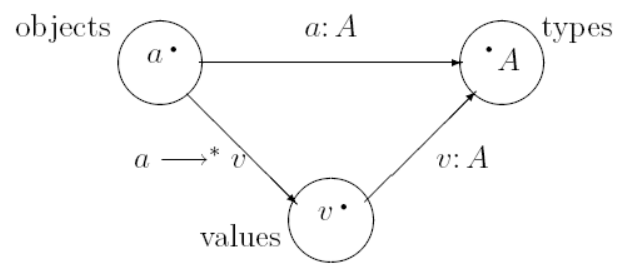
- Copredication
  - Linguistic phenomenon studied by many (Pustejovsky, Asher, Cooper, Retoré, …)
  - Dot-types in MTTs: formal proposal [Luo 2009] (*), implementation [Xue & Luo 2012] and copredication with quantification [Chatzikyriakidis & Luo 2018]
  - Linguistic feature difficult, if not impossible, to find satisfactory treatment in a CNs-as-predicates framework. (For a mereological one, see [Gotham16].)
- Anaphora analysis/resolution via Σ-types
  - [Sundholm 1986, Ranta 1994] in Martin-Löf's type theory
- Linguistic coercions via coercive subtyping [Asher & Luo 2012]
- Several recent developments
  - Propositional forms of judgemental interpretations [Xue et al (NLCS18)]
  - CNs as setoids [Chatzikyriakidis & Luo (Oslo Studies in Language 2018)]
  - (later today) MTT-sem in Martin-Löf's TT with h-logic [Luo (LACompLing18)]
  - (Wednesday) Event semantics in MTT-framework [Luo & Soloviev (WoLLIC17)]

II. Universes and Π-polymorphism

- Example for a first 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_{run}→Prop)→(A_{run}→Prop), where A_{run} is domain for run.
  - Other verbs? Adjectives? Generically? One type for all?
- Π-types for polymorphism come for a rescue: (*)
  quickly : ΠA:CN. (A→Prop)→(A→Prop)
- Q: What is CN? A: CN is a universe of types (ie, of CNs).

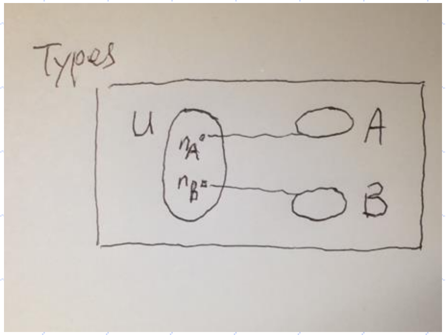
Universes in type theory

- Objects and types:
  - Two worlds connected by a:A.
  - Types collect objects into totalities.



- What if we want to collect some types into a totality?
  - Collecting (the names of) some types into a new type.
  - E.g., common nouns are types; Can we have a type CN whose objects are the types that interpret common nouns?
  - Yes, we need a universe CN.

- Martin-Löf introduced the notion of universe (1973).
- A universe is a type of (names of) types.



- Notes on Π-quantification
  - Let U be a universe.
  - We can quantify over U to have, e.g.,

    ΠX:U. … …

    Functions of this type is polymorphic. (c.f., quickly)
  - Let Type be the collection of all types. One cannot use Π to quantify over Type to form type ΠX:Type.…, because Type itself cannot be a type – otherwise, logical paradox.

- Examples in mathematics
  - Type theory as foundation of math, one needs to define type-valued functions.
  - f(n) = Nat x … x Nat (n times)
  - Universe containing Nat is needed because a function's codomain must be a type (the universe in this case; it cannot be Type – paradox).
- Examples in MTT-semantics – today
  - Linguistic universes (CN, LType)
  - Logical universes (Prop in UTT, $PROP_U$ in $MLTT_h$)

III. Linguistic universes

- Let's start by reviewing CN
  - Universe of (interpretations of) common nouns
  - CN : Type
  - Let A : Type be the interpretation of some common noun.
  - Then, n_A : CN (name of A) and $T_{CN}(n_A) = A$.
  - Omitting T_{CN} and identifying n_A with A, we have A : CN.
- Example (review): predicate-modifying adverbs
  - Montague: quickly : $(e \rightarrow t) \rightarrow (e \rightarrow t)$
  - MTT-semantics: quickly : $\Pi A{:}CN.\ (A \rightarrow Prop) \rightarrow (A \rightarrow Prop)$
    - "run quickly" – quickly(A_{run} , run) : $A_{run} \rightarrow Prop$
    - "begin quickly" – quickly(A_{begin} , begin) : $A_{begin} \rightarrow Prop$

Modelling subsective adjectives

- Nature of such adjectives
  - Their meanings are dependent on the nouns they modify.
  - Eg, "a large mouse" is not a large animal
- Our proposal:
  - large : ΠA:CN. (A→Prop)
  - large(Mouse) : Mouse → Prop
  - [large mouse] = Σx:Mouse. large(Mouse)(x)

skilful [CL 2014]

- If skilful : ΠA:CN. (A→Prop)
  - skilful(Doctor) : Doctor → Prop
  - [skilful doctor] = Σx:Doctor. skilful(Doctor)(x)
- But, we could also have “skilful car”. How to exclude it?
- skilful : ΠA:CN_H. (A→Prop)
  - CN_H – sub-universe of CN of subtypes of Human

$$\frac{A : CN \qquad A \leq Human}{A : CN_H}$$

  - Then, under the above typing for skilful with CN_H ,
    - skilful(Doctor) : Doctor → Prop
    - skilful(Car) is ill-typed (and excluded).

Another example – type of quantifiers [LL 2014]

- Generalised quantifiers
  - Examples: some, three, a/an, all, …
  - In sentences like: "Some students work hard."
- With Π-polymorphism, the type of binary quantifiers is:

  $\Pi A{:}CN.\ (A \rightarrow Prop) \rightarrow Prop$

  For Q of the above type

  $N : CN,\ \ V : N \rightarrow Prop \ \ \Rightarrow \ \ Q(N,V) : Prop$

  E.g., Student : CN, work_hard : Human→Prop

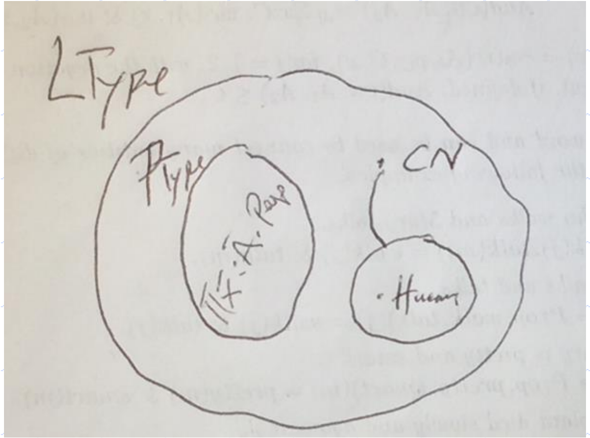
  ➔ Some(Student,work_hard) : Prop

Note: the above only works because Student ≤ Human.

LType: universe for modelling coordination [CL12]

- Examples of conjoinable types
  - John walks and Mary talks. (sentences)
  - John walks and talks. (verbs)
  - Mary is pretty and smart. (adjectives)
  - The plant died slowly and agonizingly. (adverbs)
  - Every student and some professors came. (quantified NPs)
  - Some but not all students got an A. (quantifiers)
  - John and Mary went. (proper names)
  - A friend and colleague came. (CNs)
  - … …
- Question: can we consider coordination <u>generically</u>?

- LType – the universe of "linguistic types", with formal rules in the next slide.
  - PType ≤ LType
  - CN ≤ LType



- Example types in LType:
  - Prop (type of sentences)
  - Type of predicate-modifying adverbs:
    $\Pi A{:}CN.\ (A \rightarrow Prop) \rightarrow (A \rightarrow Prop)$
  - Type of quantifiers: $\Pi A{:}CN.\ (A \rightarrow Prop) \rightarrow Prop$
  - Types such as Human that interpret CNs
  - Universe CN of common nouns

$$\frac{}{PType : Type} \qquad \frac{}{Prop : PType} \qquad \frac{A : LType \quad P(x) : PType\ [x{:}A]}{\Pi x{:}A.P(x) : PType}$$

$$\frac{}{LType : Type} \qquad \frac{}{\mathrm{CN} : LType} \qquad \frac{A : \mathrm{CN}}{A : LType} \qquad \frac{A : PType}{A : LType}$$

Fig. 1. Some (not all) introduction rules for *LType*.

- Then, coordination can be considered generically:
  - Every (binary) coordinator such as <u>And</u> is of type

$$\Pi A:LType.\ A \rightarrow A \rightarrow A$$

- We can then type the coordination examples.
  - Mary is pretty and smart.
    - And(Human→Prop, pretty, smart)(m)
  - Every student and some professors came.
    - And((Human→Prop)→Prop, every(Student), some(Professor))(come)
  - John and Mary went.
    - go(And(Human, j, m))

- Now, although generic typing is OK, but what do these And-terms mean?
- For distributive readings, do we have:
  - And(pretty, smart)(m) ⇔ pretty(m) & smart(m)
  - And(every(Student), some(Professor))(come)

    ⇔ every(Student,come) & some(Professor,come)
  - go(And(j, m)) ⇔ go(j) & go(m)
- This was not dealt with in [CL12]. We now give meaning so that, for distributive readings, such equivalences become true (see next slide).

- The distributive meaning of And(A) : A→A→A, by case analysis of type A:

  - $A \equiv \Pi x_1{:}A_1 ... \Pi x_n{:}A_n.Prop\ (n \in \omega)$. *Then, for any* $f, g : A$,

  $$And(A, f, g)(x_1, ..., x_n) =_{df} f(x_1, ..., x_n)\ \&\ g(x_1, ..., x_n)\ :\ Prop.$$

  When $n = 0$, *the above definition reduces to* $And(Prop, P, Q) = P\ \&\ Q : Prop$.

  - $A : \textsc{cn}$. *Then, for* $a_1, a_2 : A$ *and* $P : A \to Prop$,

  $$P(And(A, a_1, a_2)) =_{df} P(a_1)\ \&\ P(a_2).$$

  - $A \equiv \textsc{cn}$. *Then, for* $A_1, A_2 : \textsc{cn}$ *and for* $C : \textsc{cn}$ *such that* $A_i \leq C\ (i = 1, 2)$,

  $$And(\textsc{cn}, A_1, A_2) =_{df} \Sigma x{:}C.\ \textsc{is}_C(A_1, x)\ \&\ \textsc{is}_C(A_2, x)$$

(Tech details omitted in this talk.)

IV. Logical universes for MTT-semantics

- Logics in MTTs
  - Propositions as types – in judgement “t : T”, T can be a proposition and, in that case, t is a proof of T and T is true.
- Proof irrelevance
  - Any two proofs of the same proposition are the same.
  - To have adequate MTT-semantics, proof irrelevance needs be enforced in the underlying type theory.

Examples in semantics

- Identity criteria for modified CNs [Luo (LACL 2012)]
  - A handsome man is interpreted as a pair (m,p) of a Σ-type Σx:Man.handsome(x).
  - Two handsome men are the same iff they are the same man ➔ proof irrelevance.
- Counting (the same problem as above) [Tanaka 15]
  - Any farmer who owns donkeys beat <u>most</u> of them.
  - Counting pairs incorrectly takes proofs into account.
  - Tanaka proposed a solution (ad hoc and complicated).
  - I believe proof irrelevance provides a clean/easy solution.

Logic in impredicative type theory UTT

- HOL in UTT
  - Prop – type of all logical propositions
    - Prop is an internal totality (c.f. t in Montague's semantics).
    - Eg, a predicate over A is of type $A \to Prop$.
  - $\Pi x{:}A.P(x)$: Prop for any type A and any predicate P. (*)
    - Other logical operators ($\wedge$, $\neg$, $\exists$, …) can all be defined by Π.
    - For example, $P \wedge Q = \Pi X{:}Prop.(P \to Q \to X) \to X$.
- UTT for MTT-semantics
  - UTT – employed in development of MTT-semantics.
  - Proof irrelevance can be enforced:

$$\frac{\Gamma \vdash P : Prop \quad \Gamma \vdash p : P \quad \Gamma \vdash q : P}{\Gamma \vdash p = q : P}$$

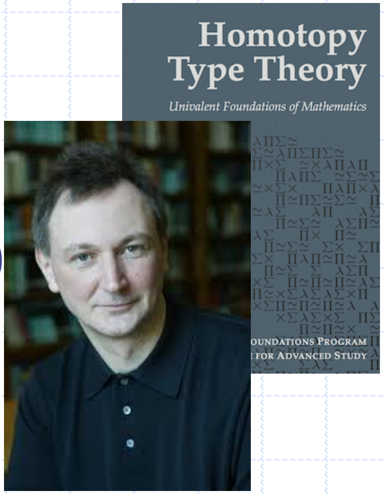
Martin-Löf's type theory with PaT logic (MLTT/PaT)

- Martin-Löf's type theory for formal semantics
  - Sundholm, Ranta & many others (all use MLTT/PaT)
- PaT logic in MLTT
  - Types and propositions are identified: types = propositions!
    - ∏/∀, Σ/∃, x/∧, +/∨, →/⊃, A→∅/¬A, … (non-standard first-order logic)
  - There is no type of all propositions (otherwise, paradox)
    - Could only approximate predicates by means of predicative universes.

Problem: Cannot have proof irrelevance in MLTT/PaT.

- In MLTT/PaT, proof irrelevance would mean that every type collapses (into empty/singleton types)! Obviously absurd.
- So, MLTT/PaT is inadequate for MTT-semantics.

MLTT_h: Extension of MLTT with H-logic



- H-logic (Voevodsky in HoTT)
  - A mere proposition is a type with at most one object. (In symbols, isProp(A) = $\prod x,y{:}A.(x=y)$.)
  - Logical operators, examples (see next slide):
    - $P \supset Q = P \to Q$ and $\forall x{:}A.P = \prod x{:}A.P$
    - $P \vee Q = [P+Q]$ and $\exists x{:}A.P = [\Sigma x{:}A.P]$

    where [_] is propositional truncation, a <u>proper</u> extension.
- MLTT_h = MLTT + h-logic (adequate for MTT-sem)
  - Proof irrelevance is “built-in” in h-logic (by definition).
  - $\text{PROP}_U = \Sigma(U, \text{isProp})$ ($= \Sigma x{:}U.\text{isProp}(x)$)
  - Note: MLTT_h does not have univalence or other HITs.
  - Details in the short paper in LACompLing18 proceedings.

- Truncation [A] is a proper extension of MLTT.
  - For any type A, stipulate existence of truncation type [A]:

$$\frac{a : A}{|a| : [A]} \qquad \frac{a : [A] \quad b : [A]}{p(a,b) : Id_A(a,b)}$$

  - Proper extension – the 2nd rule stipulates that [A] is mere.
- Only some logical operations preserve truncations:
  - If P and Q are mere propositions, so is P→Q.
  - But, for some mere propositions P and Q, P+Q is not mere.
  - That's why one needs truncations – [P+Q] is always mere!
- Remark: Canonicity fails to hold in this extension.

Concluding summary

- Universes in type-theoretic semantics
- Linguistic universes
  - CN and Π-polymorphism – a powerful tool in constructing formal semantics
  - LType – coordination typing plus new semantic development
- Logical universes in the underlying type theories
  - Prop in impredicative UTT
  - $MLTT_h$ – MLTT extended with h-logic, can be employed adequately for MTT-semantics. (MLTT/PaT is inadequate.)

Non-committal Adjectives

- Let A be a non-committal adjective and h : Human.
  - alleged, predicted, arguable, … (human agents) and others
- $\Sigma_{h,A}$: Fin(n)→Prop
  - A=alleged/predicted ➔ $\Sigma_{h,A}$ = h's allegations/predictions.
- $H_{h,A}(P) = \exists i{:}Fin(n).\ \Sigma_{h,A}(i) =_{Prop} P$
  - A=alleged/predicted ➔ $H_{h,A}(P)$ = h alleged/predicted P.
- John is an alleged criminal.
  - $\exists h{:}Human.\ H_{h,alleged}$(John is a criminal),
    - where [John is a criminal] = IS_{Human}(Criminal, j).

Π-types/∀-propositions

$$(\Pi_T)\quad \frac{\Gamma \vdash A\ type \quad \Gamma,\ x : A \vdash B[x]\ type}{\Gamma \vdash \Pi x : A.B[x]\ type}$$

$$(\Pi_P)\quad \frac{\Gamma \vdash A\ type \quad \Gamma,\ x : A \vdash P[x]\ prop}{\Gamma \vdash \Pi x : A.P[x]\ prop}$$

$$(\lambda)\quad \frac{\Gamma,\ x : A \vdash b : B}{\Gamma \vdash \lambda x : A.b : \Pi x : A.B[x]}$$

$$(app)\quad \frac{\Gamma \vdash f : \Pi x : A.B[x] \quad \Gamma \vdash a : A}{\Gamma \vdash f(a) : B[a]}$$

Π_T for Π-types and Π_P for universal quantification

Logical operators in, eg, UTT

$$
\begin{aligned}
\forall x{:}A.P[x] &=_{df} \Pi x{:}A.P[x] \\
P_1 \supset P_2 &=_{df} \forall x{:}P_1.P_2 \\
\mathbf{true} &=_{df} \forall X{:}Prop.\ X \supset X \\
\mathbf{false} &=_{df} \forall X{:}Prop.X \\
P_1 \,\&\, P_2 &=_{df} \forall X{:}Prop.\ (P_1 \supset P_2 \supset X) \supset X \\
P_1 \vee P_2 &=_{df} \forall X{:}Prop.\ (P_1 \supset X) \supset (P_2 \supset X) \supset X \\
\neg P_1 &=_{df} P_1 \supset \mathbf{false} \\
\exists x{:}A.P[x] &=_{df} \forall X{:}Prop.\ (\forall x{:}A.(P[x] \supset X)) \supset X.
\end{aligned}
$$

Definition 3.7.1. We define **traditional logical notation** using truncation as follows, where P and Q denote mere propositions (or families thereof):

$$
\begin{aligned}
\top &:\equiv \mathbf{1} \\
\bot &:\equiv \mathbf{0} \\
P \wedge Q &:\equiv P \times Q \\
P \Rightarrow Q &:\equiv P \rightarrow Q \\
P \Leftrightarrow Q &:\equiv P = Q \\
\neg P &:\equiv P \rightarrow \mathbf{0} \\
P \vee Q &:\equiv \| P + Q \| \\
\forall (x : A).\, P(x) &:\equiv \prod_{x:A} P(x) \\
\exists (x : A).\, P(x) &:\equiv \left\| \sum_{x:A} P(x) \right\|
\end{aligned}
$$