CNs as Types: HITs in Type-Theoretical Semantics*

Zhaohui Luo
Dept of Computer Science
Royal Holloway, Univ of London

* CNs – Common Nouns

* HITs – Higher Inductive Types (recent development in HoTT)

Introduction (summary)

- Modern (v.s. Simple) Type Theories
  - Predicative (MLTT) and impredicative (pCIC & UTT)
  - MTT-semantics is in Montague-style but based on MTTs.
- CNs-as-types: historical developments
  - Mönnich & Sundholm (mid-80s): Σ-types for donkey sentences
  - Ranta (1994): multiple categorization problem
  - Luo (2009): (coercive) subtyping solves the problem.
- CNs-as-setoids (A,R) – why not quotient types A/R?
  - CNs have identity criteria (Geach 1962) & in general ICs are needed.
  - CNs-as-setoids in (Luo 2012) since, then, quotient types as extensional constructs were still not available in intensional TTs.
  - Now, quotient types are available as HITs for intensional TTs! (eg, in cubical type theory (Coquand et al TYPES15/LICS18))

This talk

I. CNs as types/setoids: an introduction
(paradigm, problem, difficulty)
II. Type theories as foundational languages
(why properties such as canonicity are essential)
III. Quotient types as HITs
(quotient types solve our problem and HITs help to justify)

I. CNs as types in type-theoretical semantics

- CNs as predicates in Montague semantics in simple type theory
  - cat : $\mathbf{e} \to \mathbf{t}$
  - black cat = $\lambda x{:}\mathbf{e}.\ \text{cat}(x) \wedge \text{black}(x)$
- CNs as types in MTT-semantics (in Modern Type Theories)
  - Cat : Type (just like **e** being a type)
  - Black-cat = $\Sigma x{:}\text{Cat}.\text{black}(x)$, where black : $\text{Cat} \to \text{Prop}$
- Remarks (cf, MTT-semantics for adj mod by Chatzikyriakidis & Luo)
  - Montague uses the "single-sorted" simple TT (discounting **t**), while MTTs may be considered as "many-sorted".
  - Rich typing in representing CNs, besides Σ in the above
    - Π-polymorphism for subsective adjectival modification
    - Disjoint union types for privative adjectival modification

Criteria of Identity (IC)

- Example (c.f., Gupta 1980)
  (1) EasyJet has transported 1 million passengers in 2010.
  (2) EasyJet has transported 1 million persons in 2010.
  - (1) doesn't imply (2): the CNs 'passenger' & 'person' have different identity criteria.
- Criteria of identity have been studied in
  - Philosophy (Geach 1962): Nouns have identity criteria (eg, his example of "cat on mat").
  - Linguistics (Baker 2003): Identity criteria are special for nouns (not for other categories such as verbs/adjectives/…).
  - Constructive math (Bishop 1967): Sets are 'presets' with equivalence, that is, setoids (A,=).

Misc remarks

- ICs enable individuation, counting, quantification, ...
- Representation-wise, a passenger could be a pair of a person together with a journey made by the person:

  $$\text{Passenger} = \Sigma p : \text{Person}.\ J(p)$$

- In Chinese, noun phrase 人次 captures this (sum of each time's numbers – such a term in English?)

Discrepancy and quotient types

- In type-theoretical semantics
  - CNs as types: Mönnich (1985), Sundholm (1986), Ranta (1994)
  - CNs as types/setoids: (Luo 2012, Chatzikyriakidis & Luo 2018)
- CNs as setoids
  - CNs as types – what we usually say. But, in general, CNs as setoids – pairs (A,R), which are not types!
  - In most cases, we get away by CNs-as-types by ignoring IC.
    - Example: Two men are the same iff they are the same humans – so, Man just inherits its equality from Human.
- Why not quotient types A/R?
  - It is an extensional construct, causing problems … & now …

II. Type theories as foundational languages

- Relationship between logic and set/type theory

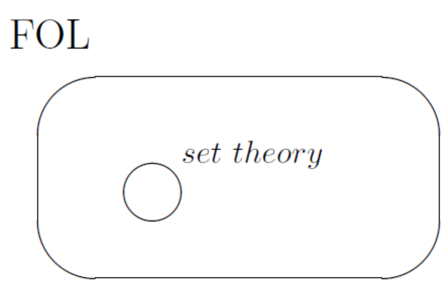

Figure 1: Set theory – a theory in first-order logic

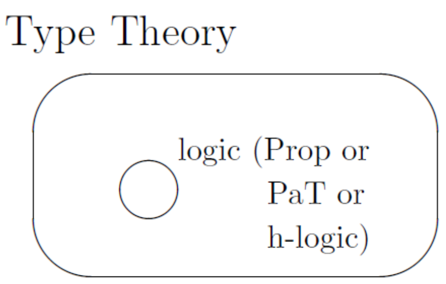

Figure 2: Logic is a part of type theory

Inductive types: an example

- Example to illustrate a key issue
- Peano axioms: logical theory for natural numbers.
  [N is a predicate and n∈N stands for N(n)]

$$(P1)\ \ 0 \in N$$
$$(P2)\ \ \forall x.\ x \in N \Rightarrow succ(x) \in N$$
$$(P3)\ \ \forall x, y.\ x, y \in N \wedge succ(x) = succ(y) \Rightarrow x = y$$
$$(P4)\ \ \forall x.\ x \in N \Rightarrow 0 \neq succ(x)$$
$$(P5)\ \ \forall P.\ P(0) \wedge [\forall x.\ x \in N \wedge P(x) \Rightarrow P(succ(x))] \Rightarrow \forall z.\ z \in N \Rightarrow P(z)$$

- Martin-Löf's idea
  - Inductive types as "computational theories"
  - Example – Nat, the type of natural numbers

Rules for Nat

- Formation and introduction rules

$$\frac{}{Nat\ type} \qquad \frac{}{0 : Nat} \qquad \frac{n : Nat}{succ(n) : Nat}$$

- Elimination rule

$$\frac{\Gamma, z : Nat \vdash C(z)\ type \quad \Gamma \vdash n : Nat \quad \Gamma \vdash c : C(0) \quad \Gamma, x : Nat, y : C(x) \vdash f(x,y) : C(succ(x))}{\Gamma \vdash \mathcal{E}_{Nat}(c, f, n) : C(n)}$$

- Computation rules (omitted)

- Notes:
  - Introduction rules specify canonical nats (0 and succ(n)).
  - Elimination rule is Nat-induction + primitive recursion.
- Why is a computational theory adequate?
  - All Peano axioms are either rules or provable.
  - Canonicity is key: e.g., every nat is equal to a canonical nat (shown in meta-theory – normalisation property).
  - E.g., contextual “axioms” are problematic – they may lead to failure of canonicity (and lose foundational adequacy).
  - Extensional constructs (eg, quotients) – also a problem.

III. Quotient types as HITs

- Extensional constructs – example of inadequacy in traditional intensional TTs
  - Hofmann's thesis on extensional constructs (1995)
  - Quotients – typical example of extensional constructs
  - Canonicity fails with quotient types – see, e.g., (Li 2014)
- Aside on Extensional Type Theory (Martin-Löf 1982/1984)
  - Identifying propositional Id(a,b) with definitional a=b.
  - A//R in Nuprl's type theory (NuPRL 1986)
  - Properties such as canonicity/SN/decidability fail to hold.
  - Adequacy of ETT as a foundational language is questionable.

Quotients as higher inductive types

- Basic idea of HITs:
  - Ordinary induction is only about "points" (0, succ(n)).
  - Higher induction extends it to "equalities/paths".
- Quotients "A/R" – a typical example:

```
|_| : A -> A/R
∀x,y:A. R(x,y) -> |x|=|y|
∀x,y:A/R.∀p,q:x=y. p=q     [ 'set truncation' ]
```

  - Intuitively, `A/R` consists of the "equivalence classes" `|a|` for `a:A`.
  - If `R` is an equivalence, then `A/R` is effective (`|x|=|y|->R(x,y)`).
  - So, a setoid pair (A,R) can be represented as a type A/R.
- Remark: Quotient types in "old" TTs were problematic (non-canonicity) – so real progress in cubical TT (next page).

Cubical type theory (Coquand …)

- Cubical type theory
  - Research started in 2012-13 at Princeton, by Coquand et al, when Voevodsky had the conjecture: canonicity holds.
  - Coquand et al TYPES15/LICS18; Vezzosi et al 2021.
- Good news:
  - Univalence is a theorem.
  - HITs are computationally justified.
  - Canonicity holds – a big step forward!
- Notes: Several research topics, including:
  - Independent understanding of HITs? …

Summary

- CNs have identity criteria
  - So, CNs are not just types, but setoids in general.
- Quotient types are “better” than setoids.
  - Quotient types as HITs are adequate.
- So, CNs as types.