Type Theory in Formal Semantics

Dilip Ninan | Tufts University | `dilip.ninan@tufts.edu`
October 2022

1. Natural language syntax

Phrase Structure Rules

(1) S → DP VP

(2) S → S CoordP

(3) CoordP → Coord S

(4) VP → V (DP|AP|PP|NegP)

(5) NegP → Neg VP|AP

(6) AP → A (PP)

(7) DP → D (NP)

(8) NP → N (PP)

(9) NP → A NP

(10) PP → P DP

Lexicon

Coord: *and*, *or*

Neg: *thin*, *tall*, *happy*

V: *smiled, laughed, loves, hugged, is, did*

A: *Swedish, happy, kind, proud*

N: *singer, drummer, musician*

D: *the, a, every, some, no*

D: *Mary, John, Sue, everybody, somebody, nobody*

P: *of, with*

2. The simply typed lambda calculus

2.1. Syntax

Definition 1. The set T of *types* is smallest set such that:

(1) $e, t \in T$, and

(2) if $\sigma, \tau \in T$, then $(\sigma, \tau) \in T$.

Definition 2. Any language $\mathcal{L}$ of the typed lambda calculus consists of the following:

(1) For each type σ, a countably infinite set VAR_σ of variables of type σ.

(2) For each type σ, a set CON_σ of constants of type σ.

(3) The lambda operator: λ.

(4) The connectives: $\neg, \vee, \wedge, \rightarrow, \equiv$.

(5) The quantifiers: $\forall$, $\exists$.

(6) The identity symbol: $=$.

(7) Left and right parentheses.

Definition 3. Let σ and τ be arbitrary types.

(1) If $\alpha \in VAR_\sigma$, α is an expression of type σ.

(2) If $\alpha \in CON_\sigma$, α is an expression of type σ.

(3) If α is an expression of type σ, and β an expression of type (σ, τ), then $(\beta(\alpha))$ is an expression of type τ.

(4) If α is an expression of type τ and v a variable of type σ, then $(\lambda v.\alpha)$ is an expression of type (σ, τ).

(5) If ϕ and ψ are expressions of type t, then so are: $\neg\phi$, $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$, and $(\phi \equiv \psi)$.

(6) If ϕ is an expression of type t, and v is a variable of type σ, $\forall v\phi$ and $\exists v\phi$ are expressions of type t.

(7) If α and β are expressions of the same type σ, then $\alpha = \beta$ is an expression of type t.

Expressions of type t are *formulas*.

Rule for simplifying λ-expressions:

> β-reduction:
> If v is a variable of type τ, α an expression of type τ, and β an expression of any type σ, then $(\lambda v.\beta)(\alpha)$ may be reduced to $\beta[\alpha/v]$, so long as α contains no free occurrences of a variable that is bound in β.

Here, $\beta[\alpha/v]$ is the result of replacing each free occurrence of v in β with α.

If v occurs in the scope of $\forall v, \exists v$ or λv in expression α, then that occurrence of v is bound in α; otherwise, it is free.

2.2. Semantics

Definition 4. Given a set D, we define for each type ρ an associated *domain* D_ρ as follows:

(1) $D_e = D$,

(2) $D_t = \{0, 1\}$,

(3) for any types σ, τ, $D_{(\sigma,\tau)} = {D_\tau}^{D^\sigma}$, i.e. the set of all functions from D_σ into D_τ.

Definition 5. A *model for* $\mathcal{L}$ is a pair $M = (D, I)$ consisting of a non-empty set D and an interpretation function I which, for each type σ, maps each expression $\alpha \in CON_\sigma$ to an element of D_σ.

Definition 6. A variable assignment g on a model (D, I) is a function which, for each type σ, maps each variable $v \in VAR_\sigma$ to an element of D_σ. If o is an element of D, $g[o/v]$ is the variable assignment h such that $h(v) = o$ and $h(v') = g(v')$ for all variables other than v.

Definition 7. The interpretation of an expression α of type σ relative to a model $M = (D, I)$ and a variable assignment g is denoted by $[\![\alpha]\!]^{M,g}$.

(1) If α is a variable of type σ, $[\![\alpha]\!]^{M,g} = g(\alpha)$.

(2) If α is a constant of type σ, $[\![\alpha]\!]^{M,g} = I(\alpha)$.

(3) If α is an expression of type σ and β an expression of type (σ, τ), $[\![\beta(\alpha)]\!]^{M,g} = [\![\beta]\!]^{M,g}([\![\alpha)]\!]^{M,g}$.

(4) If α is an expression of type τ and v a variable of type σ, then $[\![\lambda v.\alpha]\!]^{M,g}$ is the function $f \in D_{(\sigma,\tau)}$ such that for any $o \in D_\sigma$, $f(o) = [\![\alpha]\!]^{M,g[o/v]}$.

(5) If ϕ and ψ are expressions of type t, then

  (a) $[\![\phi \wedge \psi]\!]^{M,g} = 1$ iff $[\![\phi]\!]^{M,g} = [\![\psi]\!]^{M,g} = 1$
  (b) $[\![\phi \vee \psi]\!]^{M,g} = 1$ iff $[\![\phi]\!]^{M,g} = 1$ or $[\![\psi]\!]^{M,g} = 1$
  (c) $[\![\phi \rightarrow \psi]\!]^{M,g} = 1$ iff $[\![\phi]\!]^{M,g} = 0$ or $[\![\psi]\!]^{M,g} = 1$
  (d) $[\![\phi \equiv \psi]\!]^{M,g} = 1$ iff $[\![\phi]\!]^{M,g} = [\![\psi]\!]^{M,g}$

(6) If ϕ is an expression of type t and v is a variable of any type σ, then:

  (a) $[\![\forall v \phi]\!]^{M,g} = 1$ iff for all elements $o \in D_\sigma$, $[\![\phi]\!]^{M,g[o/v]} = 1$
  (b) $[\![\exists v \phi]\!]^{M,g} = 1$ iff for some element $o \in D_\sigma$, $[\![\phi]\!]^{M,g[o/v]} = 1$

(7) If α and β are expressions of the same type σ, then $[\![\alpha = \beta]\!]^{M,g} = 1$ iff $[\![\alpha]\!]^{M,g} = [\![\alpha]\!]^{M,g}$.

Proposition 1. *For any model M, variable assignment g, variable v and expression α of type τ, and expressions β an expression of type σ:*

$[\![(\lambda v.\beta)(\alpha)]\!]^{M,g} = [\![\beta[\alpha/v]]\!]^{M,g}$*, so long as α contains no free occurrences of a variable that is bound in β.*

Definition 8. If Σ is a set of formulas and ϕ a formula, $\Sigma \vDash \phi$ iff for all models M and variable assignments g, if every $\psi \in \Sigma$ is such that $[\![\psi]\!]^{M,g} = 1$, then $[\![\phi]\!]^{M,g} = 1$.

Translating trees:

- $\rightsquigarrow$: "is translated by"
  "John" $\rightsquigarrow j_e$. (*"John" is translated by j_e*)
- Two rules for translating nodes ("composition rules"):
  (1) Non-branching Nodes (NN)
  Let α be a node whose only daughter is β.
  Then if $\beta \rightsquigarrow \beta'$, $\alpha \rightsquigarrow \beta'$.
  (2) Function Application (FA)
  Let α be a node, the set of whose daughters is $\{\beta, \gamma\}$, where $\beta \neq \gamma$.
  Then if $\beta \rightsquigarrow \beta'_{(\sigma,\tau)}$ and $\gamma \rightsquigarrow \gamma'_\sigma$, then $\alpha \rightsquigarrow \beta'(\gamma')$.