Type-Theory of Acyclic Algorithms and its Reduction Calculus, I–II

Roussanka Loukanova

Department of Algebra and Logic
Institute of Mathematics and Informatics
Bulgarian Academy of Sciences, Bulgaria

29 Jan 2021 and 5 Feb 2021

Algorithmic Semantics of L^{λ}_{ar} for acyclic computations extended to L^{λ}_{ra} for restricted computations

$$\underbrace{\text{Syntax of } L^{\lambda}_{ar} \;/\; L^{\lambda}_{ra} \; (L^{\lambda}_{r}) \Longrightarrow \underbrace{\text{Algorithmic Semantics}}_{\text{Canonical Computations}} \Longrightarrow \text{Denotations}}_{\text{Algorithmic and Denotational Semantics of } L^{\lambda}_{ar} \,/\, L^{\lambda}_{ra} (L^{\lambda}_{r})}$$

- Denotational semantics of L^{λ}_{ar} / L^{λ}_{ra} (L^{λ}_{r}):
  den(A) by structural induction on $A \in$ Terms:
- Algorithmic semantics of L^{λ}_{ar} / L^{λ}_{ra} (L^{λ}_{r}):
  determined by canonical terms via the reduction calculi
  1. Every $A \in \text{Terms}_\sigma$ is reduced to its canonical form cf$(A) \in \text{Terms}_\sigma$:
  $$A \Rightarrow_{\text{cf}} \text{cf}(A) \tag{1}$$
  2. For every algorithmically meaningful $A \in \text{Terms}_\sigma$, cf$(A)$ determines the algorithm alg(A) for computing den(A)
  - L^{λ}_{ar} introduced by Moschovakis [2], 1989, [3], 2006
  - L^{λ}_{ra} introduced by Loukanova [1]

Types: $\sigma :\equiv \mathsf{e} \mid \mathsf{t} \mid \mathsf{s} \mid (\tau_1 \to \tau_2)$
For all $\tau \in$ Types:

$\mathsf{Consts}_\tau = \{\mathsf{c}_0^\tau, \mathsf{c}_1^\tau, \dots, \mathsf{c}_{k_\tau}^\tau\}$

$\mathsf{Vars}_\tau = \mathsf{PureV}_\tau \cup \mathsf{RecV}_\tau, \quad \mathsf{PureV}_\tau \cap \mathsf{RecV}_\tau = \varnothing$

$\mathsf{PureV}_\tau = \{v_0^\tau, v_1^\tau, \dots\}, \ \ \mathsf{MemoryV}_\tau = \mathsf{RecV}_\tau = \{p_0^\tau, p_1^\tau, \dots\}$

Terms of $\mathrm{L}_{\mathrm{ar}}^\lambda$ ($\mathrm{L}_{\mathrm{r}}^\lambda$):

$$\mathsf{A} :\equiv \mathsf{c}^\tau : \tau \mid x^\tau : \tau \quad (\text{for } \mathsf{c}^\tau \in \mathsf{Consts}_\tau,\ x^\tau \in \mathsf{PureV}_\tau \cup \mathsf{RecV}_\tau\ ) \tag{2a}$$

$$\mid \mathsf{B}^{(\sigma\to\tau)}(\mathsf{C}^\sigma) : \tau \tag{2b}$$

$$\mid \lambda(v^\sigma)\,(\mathsf{B}^\tau) : (\sigma \to \tau) \quad (\text{for } v^\sigma \in \mathsf{PureV}_\sigma\ ) \tag{2c}$$

$$\mid \big[\mathsf{A}_0^{\sigma_0} \text{ where } \{\, p_1^{\sigma_1} := \mathsf{A}_1^{\sigma_1}, \dots, p_i^{\sigma_i} := \mathsf{A}_i^{\sigma_i}, \dots, p_n^{\sigma_n} := \mathsf{A}_n^{\sigma_n} \,\}\big] : \sigma_0 \tag{2d}$$

$$\mid \big[\mathsf{A}_0^{\sigma_0} \text{ such that } \{\, \mathsf{C}_1^{\tau_1}, \dots, \mathsf{C}_m^{\tau_m} \,\}\big] : \sigma_0' \tag{2e}$$

- $\mathsf{B}, \mathsf{C} \in$ Terms, $\ p_i^{\sigma_i} \in \mathsf{RecV}_{\sigma_i}$, $A_i^{\sigma_i} \in \mathsf{Terms}_{\sigma_i}$
  $\mathsf{C}_j^{\tau_j} \in \mathsf{Terms}_{\tau_j}$ (for propositions): $\tau_j \equiv \mathsf{t}$ or $\tau_j \equiv \widetilde{\mathsf{t}} \equiv (\mathsf{s} \to \mathsf{t})$
- Acyclicity Constraint, for $\mathrm{L}_{\mathrm{ar}}^\lambda$; without it, $\mathrm{L}_{\mathrm{r}}^\lambda$
  $\{\, p_1^{\sigma_1} := \mathsf{A}_1^{\sigma_1}, \dots, p_i^{\sigma_i} := \mathsf{A}_i^{\sigma_i}, \dots, p_n^{\sigma_n} := \mathsf{A}_n^{\sigma_n} \,\}$ is acyclic iff:
  - there is a rank: $\{p_1, \dots, p_n\} \to \mathbb{N}$ such that:
    if $p_j \in \mathsf{FreeVars}(A_i)$ then $\mathsf{rank}(p_i) > \mathsf{rank}(p_j)$

Syntax of TT of Restricted Algorithms $\mathrm{L}^{\lambda}_{\mathrm{ra}}$

$$A :\equiv \mathsf{c}^{\tau} : \tau \;\mid\; x^{\tau} : \tau \;\mid\; B^{(\sigma\to\tau)}(C^{\sigma}) : \tau \;\mid\; \lambda(v^{\sigma})\,(B^{\tau}) : (\sigma\to\tau) \quad \text{(3a)}$$

$$\mid\; \big(A_0^{\sigma_0} \text{ where } \{p_1^{\sigma_1} := A_1^{\sigma_1}, \dots, p_n^{\sigma_n} := A_n^{\sigma_n}\}\big) : \sigma_0 \quad \text{(3b)}$$

$$\mid\; \big(A_0^{\sigma_0} \text{ such that } \{\, C_1^{\tau_1}, \dots, C_n^{\tau_n} \,\}\big) : \sigma_0' \quad \text{(3c)}$$

In (3b): $p_i \in \mathsf{RecV}^{\sigma_i}$, $A_i \in \mathsf{Terms}^{\sigma_i}$ satisfy Acyclicity Constraint:

- $\{p_1^{\sigma_1} := A_1^{\sigma_1}, \dots, p_n^{\sigma_n} := A_n^{\sigma_n}\}$ is acyclic, i.e., exists a function rank: $\{p_1, \dots, p_n\} \to \mathbb{N}$
  s.th. if p_j occurs freely in A_i, then $\mathrm{rank}(p_i) > \mathrm{rank}(p_j)$

In (3c): For each $i = 1, \dots, n$,
$\tau_i \equiv \mathsf{t}$ (truth values) or $\tau_i \equiv \widetilde{\mathsf{t}} \equiv (\mathsf{s} \to \mathsf{t})$ (state dependent truth values)

$$\sigma_0' \equiv \begin{cases} \sigma_0, & \text{if } \tau_i \equiv \mathsf{t}, \text{ for all } i \in \{1, \dots, n\} \\ \sigma_0 \equiv (\mathsf{s}\to\sigma), & \text{if for some } \sigma \in \mathsf{Types},\ \sigma_0 \equiv (\mathsf{s}\to\sigma) \\ \widetilde{\sigma_0} \equiv (\mathsf{s}\to\sigma_0), & \text{otherwise, i.e.,} \\ & \text{if } \tau_i \equiv \widetilde{\mathsf{t}}, \text{ for some } i \in \{1,\dots,n\}, \text{ and} \\ & \quad \text{there is no } \sigma \text{ s.th. } \sigma_0 \equiv (\mathsf{s}\to\sigma) \end{cases} \quad (4)$$

Abbreviations

- Carnap's Intensions, the type of state dependent objects of type σ:

$$\widetilde{\tau} \equiv (\mathsf{s} \to \tau), \quad \text{for } \tau \in \text{Types} \tag{5}$$

- Sequences

$$\overrightarrow{X} \equiv X_1, \dots, X_n \quad (n \geq 0) \tag{6a}$$

$$\text{of terms: } X_i \in \text{Terms, for all } i \in \{1, \dots, n\} \tag{6b}$$

$$\text{of types: } X_i \in \text{Types, for all } i \in \{1, \dots, n\} \tag{6c}$$

- Abbreviated sequences of mutually recursive assignments:

$$\overrightarrow{p} := \overrightarrow{A} \equiv [p_1 := A_1, \ \dots, \ p_n := A_n] \quad (n \geq 0) \tag{7}$$

- Abbreviated restrictor operator (such that $\equiv$ s.t.) and terms:

$$\big(A_0 \text{ such that } \{ C_1, \dots, C_n \}\big) \tag{8a}$$

$$\equiv \big(A_0 \text{ s.t. } \{ C_1, \dots, C_n \}\big) \tag{8b}$$

$$\equiv \big(A_0 \text{ such that } \{ \overrightarrow{C} \}\big) \equiv \big(A_0 \text{ such that } \overrightarrow{C}\big) \tag{8c}$$

Denotational Semantics of L_{ar}^{λ}

A standard semantic structure is a tuple $\mathfrak{A}(\text{Consts}) = \langle \mathbb{T}, \mathcal{I} \rangle$ that satisfies the following conditions:

- $\mathbb{T} = \{\mathbb{T}_\sigma \mid \sigma \in \text{Types}\}$ is a frame of typed objects
  $\{0, 1, er\} \subseteq \mathbb{T}_\mathsf{t} \subseteq \mathbb{T}_\mathsf{e}$ $(er_\mathsf{t} \equiv er_\mathsf{e} \equiv er \equiv error)$
  $\mathbb{T}_\mathsf{s} \neq \varnothing$ (the domain of *states*)
  $\mathbb{T}_{(\tau_1 \to \tau_2)} = (\mathbb{T}_{\tau_1} \to \mathbb{T}_{\tau_2}) = \{ f \mid f \colon \mathbb{T}_{\tau_1} \to \mathbb{T}_{\tau_2} \}$ (standard str.)
  $er_\sigma \in \mathbb{T}_\sigma$, for every $\sigma \in \text{Types}$ (designated typed errors)
- $\mathcal{I} \colon \text{Consts} \longrightarrow \cup \mathbb{T}$ is a typed *interpretation function*:
  $\mathcal{I}(\mathsf{c}) \in \mathbb{T}_\sigma$, for every $\mathsf{c} \in \text{Consts}_\sigma$
- $\mathfrak{A}$ is associated with the set of the typed variable valuations G:

$$G = \{g \mid g \colon \text{PureV} \cup \text{RecV} \longrightarrow \bigcup \mathbb{T} \text{ and, for every } X \in \text{Vars}_\sigma, \ g(X) \in \mathbb{T}_\sigma\} \tag{9}$$

The denotation function of L_{ar}^{λ} (to be continued)

- We assume a given $\mathfrak{A}$, and write den $\equiv$ den$^{\mathfrak{A}}$
- There is a unique function, called the *denotation function*:
  $\text{den}^{\mathfrak{A}} : \text{Terms} \longrightarrow \{ f \mid f : G \longrightarrow \cup \mathbb{T} \}$
  defined by recursion on the structure of the terms, by (D1)–(D5)

(D1)
1. $\text{den}(x)(g) = g(x)$, for every $x \in$ Vars
2. $\text{den}(\mathsf{c})(g) = \mathcal{I}(\mathsf{c})$, for every $\mathsf{c} \in$ Consts

(D2) $\text{den}(A(B))(g) = \text{den}(A)(g)(\text{den}(B)(g))$

(D3) $\text{den}(\lambda x(B))(g)(a) = \text{den}(B)(g\{x := a\})$, for every $a \in \mathbb{T}_\tau$

The denotation function for the recursion terms (continuation) (to be continued)

(D4) $\text{den}(A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\})(g) =$
$\text{den}(A_0)(g\{p_1 := \overline{p}_1, \ldots, p_n := \overline{p}_n\})$

where $\overline{p}_i \in \mathbb{T}_{\tau_i}$ are defined by recursion on $\text{rank}(p_i)$:

$$\overline{p_i} = \text{den}(A_i)(g\{p_{k_1} := \overline{p}_{k_1}, \ldots, p_{k_m} := \overline{p}_{k_m}\})$$

given that $p_{k_1}, \ldots, p_{k_m}$ are all of the recursion variables $p_j \in \{p_1, \ldots, p_n\}$, s.t. $\text{rank}(p_j) < \text{rank}(p_i)$.

Intuitively:

- $\text{den}(A_1)(g), \ldots, \text{den}(A_n)(g)$ are computed recursively, by $\text{rank}(p_i)$, and stored in p_i, $0 \leq i \leq n$
- the denotation $\text{den}(A_0)(g)$ may depend on the values stored in $p_1, \ldots, p_n$

The denotation function for the restrictor terms (continuation) (to be continued)

(D5)

Case 1: for all $i \in \{1, \dots, n\}$, $C_i \in \mathsf{Terms}_{\mathsf{t}}$
For every $g \in G$:

$$\mathrm{den}\left(A_0^{\sigma_0} \text{ s.t. } \{\overrightarrow{C}\}\right)(g) = \begin{cases} \mathrm{den}(A_0)(g), & \text{if, for all } i \in \{1, \dots, n\}, \\ & \quad \mathrm{den}(C_i)(g) = 1 \\ & \\ er_{\sigma_0} & \text{if, for some } i \in \{1, \dots, n\}, \\ & \quad \mathrm{den}(C_i)(g) = 0 \text{ or} \\ & \quad \mathrm{den}(C_i)(g) = er \end{cases}$$

Case 2: for some $i \in \{1, \ldots, n\}$, $C_i : \tilde{\mathsf{t}}$, i.e., $C_i \in \mathsf{Terms}_{\tilde{\mathsf{t}}}$ (a state dependent proposition)

For every $g \in G$, and every state $s \in \mathbb{T}_{\mathsf{s}}$:

$$\mathsf{den}\left(A_0^{\sigma_0} \text{ s.t. } \{\overrightarrow{C}\}\right)(g)(s) = \begin{cases} \mathsf{den}(A_0)(g)(s), & \text{if } \mathsf{den}(C_i)(g) = 1, \\ & \quad \text{for all } i \text{ s.th. } C_i : \mathsf{t}, \text{ and} \\ & \mathsf{den}(C_i)(g)(s) = 1, \\ & \quad \text{for all } i \text{ s.th. } C_i : \tilde{\mathsf{t}}, \text{ and} \\ & \sigma_0 \equiv (\mathsf{s} \to \sigma) \\ \mathsf{den}(A_0)(g), & \text{if } \mathsf{den}(C_i)(g) = 1, \\ & \quad \text{for all } i \text{ s.th. } C_i : \mathsf{t}, \text{ and} \\ & \mathsf{den}(C_i)(g)(s) = 1, \\ & \quad \text{for all } i \text{ s.th. } C_i : \tilde{\mathsf{t}}, \text{ and} \\ & \sigma_0 \not\equiv (\mathsf{s} \to \sigma), \\ & \quad \text{for all } \sigma \in \mathsf{Types} \\ er_{\sigma_0'}(s) \text{ [alt. } er], & \text{otherwise} \end{cases}$$

Immediate terms do not carry algorithmic sense;
their denotations are by the variable valuations

Definition (The set ImT of immediate terms)

$$\mathsf{ImT}^{\tau} :\equiv X^{\tau} \mid Y^{(\tau_1 \to \cdots \to (\tau_m \to \tau))}(v_1^{\tau_1}) \ldots (v_m^{\tau_m}) \tag{10a}$$

(immediate applicative terms)

$$\mathsf{ImT}^{(\sigma_1 \to \cdots \to (\sigma_n \to \tau))} :\equiv \lambda(u_1^{\sigma_1}) \ldots \lambda(u_n^{\sigma_n}) Y^{(\tau_1 \to \cdots \to (\tau_m \to \tau))}(v_1^{\tau_1}) \ldots (v_m^{\tau_m}) \tag{10b}$$

(immediate λ-terms)

for $n \geq 0$, $m \geq 0$; $u_i, v_j \in$ PureV, $X \in$ Vars, $Y \in$ RecV

Definition (Proper terms)

$$\mathsf{PrT} = (\mathsf{Terms} - \mathsf{ImT}) \tag{11}$$

Definition (Congruence Relation, informally)

The *congruence* relation is the smallest equivalence relation (i.e., reflexive, symmetric, transitive) between $\mathrm{L}_{\mathrm{ar}}^{\lambda}$-terms, $A \equiv_{\mathsf{c}} B$, that is closed under:

1. operators of term-formation:
   - application
   - λ-abstraction
   - acyclic recursion
   - restriction term
2. renaming bound variables (pure and recursion), without causing variable collisions
3. re-ordering of the assignments within the acyclic sequences of assignments in the recursion terms
4. re-ordering of the restriction sub-terms in the restriction terms

[Congruence] If $A \equiv_{\mathsf{c}} B$, then $A \Rightarrow B$ (cong)

[Transitivity] If $A \Rightarrow B$ and $B \Rightarrow C$, then $A \Rightarrow C$ (trans)

[Compositionality]

- If $A \Rightarrow A'$ and $B \Rightarrow B'$, then $A(B) \Rightarrow A'(B')$ (ap-comp)

- If $A \Rightarrow B$, then $\lambda(u)(A) \Rightarrow \lambda(u)(B)$ (λ-comp)

- If $A_i \Rightarrow B_i$ $(i = 0, \ldots, n)$, then

$$A_0 \text{ where } \{ p_1 := A_1, \ldots, p_n := A_n \} \Rightarrow B_0 \text{ where } \{ p_1 := B_1, \ldots, p_n := B_n \} \quad \text{(wh-comp)}$$

- If $A_0 \Rightarrow B_0$ and $C_i \Rightarrow R_i$ $(i = 0, \ldots, n)$, then

$$A_0 \text{ such that } \{ C_1, \ldots, C_n \} \Rightarrow B_0 \text{ such that } \{ R_1, \ldots, R_n \} \quad \text{(st-comp)}$$

Reduction Rules (to be continued)

[Head Rule] given that no p_i occurs freely in any B_j,

$$\Big(A_0 \text{ where } \{\, \overrightarrow{p} := \overrightarrow{A} \,\}\Big) \text{ where } \{\, \overrightarrow{q} := \overrightarrow{B} \,\} \Rightarrow A_0 \text{ where } \{\, \overrightarrow{p} := \overrightarrow{A},\ \overrightarrow{q} := \overrightarrow{B} \,\} \quad \text{(head)}$$

[Bekič-Scott Rule] given that no q_i occurs freely in any A_j,

$$A_0 \text{ where } \{\, p := \Big(B_0 \text{ where } \{\, \overrightarrow{q} := \overrightarrow{B} \,\}\Big),\ \overrightarrow{p} := \overrightarrow{A} \,\} \Rightarrow A_0 \text{ where } \{\, p := B_0, \overrightarrow{q} := \overrightarrow{B},\ \overrightarrow{p} := \overrightarrow{A} \,\} \quad \text{(B-S)}$$

[Recursion-Application Rule] given that no p_i occurs freely in B,

$$\Big(A_0 \text{ where } \{\, \overrightarrow{p} := \overrightarrow{A} \,\}\Big)(B) \Rightarrow A_0(B) \text{ where } \{\, \overrightarrow{p} := \overrightarrow{A} \,\} \quad \text{(recap)}$$

Reduction Rules (to be continued)

[Application Rule] given that $B \in$ PrT is a proper term, and fresh $p \in \big[\,\text{RecV} - \big(\,\text{FV}\,(A(B)) \cup \text{BV}\,(A(B))\big)\big]$,

$$A(B) \;\Rightarrow\; \big[A(p) \text{ where } \{\, p := B \,\}\big] \tag{ap}$$

[λ-rule] given fresh $p_i' \in \big[\,\text{RecV} - \big(\,\text{FV}(A) \cup \text{BV}(A)\big)\big]$, $i = 1, \ldots, n$, for $A \equiv A_0$ where $\{\, p_1 := A_1, \ldots, p_n := A_n \,\}$

$$\begin{aligned} &\lambda(u)\Big(A_0 \text{ where } \{\, p_1 := A_1, \ldots, p_n := A_n \,\}\Big) \\ \Rightarrow &\Big[\,\lambda(u)A_0' \text{ where } \{\, p_1' := \lambda(u)A_1', \ldots, p_n' := \lambda(u)A_n' \,\}\Big] \end{aligned} \tag{λ}$$

where, for all $i = 0, \ldots, n$,

$$A_i' \equiv \big[A_i\{\, p_1 :\equiv p_1'(u), \ldots, p_n :\equiv p_n'(u) \,\}\big] \tag{16}$$

(st1) Rule given that:

- C_i $(i = 1, \dots, n,\ n \geq 0)$ are proper terms
- A_0, $\overrightarrow{I}$ (if not empty) are immediate, and
- $c_i \in \mathsf{RecV}$ $(i = 1, \dots, n)$ are fresh

$$\begin{aligned} &(A_0 \text{ such that } \{\, C_1, \dots, C_n, \overrightarrow{I} \,\}) \\ \Rightarrow\ &(A_0 \text{ such that } \{\, c_1, \dots, c_n, \overrightarrow{I} \,\}) \\ &\quad \text{where } \{\, c_1 := C_1,\ \dots, c_n := C_n \,\} \end{aligned} \tag{st1}$$

(st2) Rule given that:

- A_0, C_i $(i = 1, \dots, n,\ n \geq 0)$ are proper terms, and
- $\overrightarrow{I}$ (if not empty) are immediate
- $a_0, c_i \in \mathsf{RecV}$ $(i = 1, \dots, n)$ are fresh

$$\begin{aligned} &(A_0 \text{ such that } \{\, C_1, \dots, C_n, \overrightarrow{I} \,\}) \\ \Rightarrow\ &(a_0 \text{ such that } \{\, c_1, \dots, c_n, \overrightarrow{I} \,\}) \\ &\quad \text{where } \{\, a_0 := A_0,\ c_1 := C_1,\ \dots, c_n := C_n \,\} \end{aligned} \tag{st2}$$

Definition (Irreducible Terms)

$A \in$ Terms is *irreducible* iff

$$\text{for all } B \in \text{Terms}, \quad A \Rightarrow B \longrightarrow A \equiv_c B \tag{19}$$

Theorem (Criteria for irreducibility)

1. *Every* $A \in \text{Consts} \cup \text{Vars}$ *is irreducible*
2. $A(B)$ *is irreducible iff* B *is immediate and* A *is explicit and irreducible*
3. $\lambda(x)(A)$ *is irreducible iff* A *is explicit and irreducible*
4. $[A_0 \text{ where } \{ \overrightarrow{p} := \overrightarrow{A} \}]$ *is irreducible iff all parts* A_i *($i = 0, \ldots, n$) are explicit and irreducible*
5. $(A_0 \text{ such that } \{ \overrightarrow{C} \})$ *is irreducible iff all parts* A_0, C_i *($i = 0, \ldots, n$) are immediate*

Proof: By structural induction on terms and checking the reduction rules.

Theorem (Basic Restricted Memory Variables)

Assume that, for $n \geq 1$:

- $\overrightarrow{I_j}$ *are immediate terms, and*
- $p_i \in \mathsf{RecV}$, $i = 2, \ldots, n$, *are fresh with respect to* p_1, $\overrightarrow{I_j}$ *($j = 1, \ldots, n$)*

Then:

$$((\ldots((p_1 \text{ s.t. } \overrightarrow{I_1}) \text{ s.t. } \overrightarrow{I_2}) \ldots) \text{ s.t. } \overrightarrow{I_n}) \tag{20a}$$

$$\Rightarrow (p_n \text{ s.t. } \overrightarrow{I_n}) \text{ where } \{ p_n := (p_{n-1} \text{ s.t. } \overrightarrow{I_{n-1}}), \tag{20b}$$

$$\ldots,$$

$$p_3 := (p_2 \text{ s.t. } \overrightarrow{I_2}), \tag{20c}$$

$$p_2 := (p_1 \text{ s.t. } \overrightarrow{I_1}) \} \tag{20d}$$

Proof: by induction on n.

Basis: $n = 1$
$(p_1 \text{ s.t. } \overrightarrow{I_1}) \Rightarrow (p_1 \text{ s.t. } \overrightarrow{I_1})$ is trivially true
Induction Step: Assume (20a)–(20d), for $n \geq 1$.
Then, we reduce the term (21a) to the canonical form (21h)–(21j), by applying the reduction rules (compositionally). □

$$(\underbrace{((\ldots((p_1 \text{ s.t. } \overrightarrow{I_1}) \text{ s.t. } \overrightarrow{I_2})\ldots) \text{ s.t. } \overrightarrow{I_n})}_{p_{n+1}} \text{ s.t. } \overrightarrow{I_{n+1}}) \tag{21a}$$

by (st2)

$$\Rightarrow (p_{n+1} \text{ s.t. } \overrightarrow{I_{n+1}}) \text{ where } \{ \tag{21b}$$

$$p_{n+1} := \underbrace{((\ldots((p_1 \text{ s.t. } \overrightarrow{I_1}) \text{ s.t. } \overrightarrow{I_2})\ldots) \text{ s.t. } \overrightarrow{I_n})}\} \tag{21c}$$

by ind. hyp. and (wh-comp)

$$\Rightarrow (p_{n+1} \text{ s.t. } \overrightarrow{I_{n+1}}) \text{ where } \{ \tag{21d}$$

$$p_{n+1} := \Big[(p_n \text{ s.t. } \overrightarrow{I_n}) \text{ where } \{ \tag{21e}$$

$$p_n := (p_{n-1} \text{ s.t. } \overrightarrow{I_{n-1}}), \tag{21f}$$

$$\ldots,\ p_2 := (p_1 \text{ s.t. } \overrightarrow{I_1})\}\Big] \tag{21g}$$

by (B-S) $$\Rightarrow (p_{n+1} \text{ s.t. } \overrightarrow{I_{n+1}}) \text{ where } \{ \tag{21h}$$

$$p_{n+1} := (p_n \text{ s.t. } \overrightarrow{I_n}),\ p_n := (p_{n-1} \text{ s.t. } \overrightarrow{I_{n-1}}), \tag{21i}$$

$$\ldots,\quad p_2 := (p_1 \text{ s.t. } \overrightarrow{I_1})\} \tag{21j}$$

□

Theorem (Restricted Memory Variables)

Assume that, for $n \geq 1$:

- $\overrightarrow{C_j}$ *are proper terms, and* $\overrightarrow{I_j}$ *are immediate*
- $p_i \in \mathrm{RecV}$ *($i = 2, \ldots, n$) and* $c_j \in \mathrm{RecV}$ *($j = 1, \ldots, n$) are fresh with respect to* p_1, $\overrightarrow{C_j}$, $\overrightarrow{I_j}$ *($j = 1, \ldots, n$)*

Then:

$$((\ldots((p_1 \text{ s.t. } \{\overrightarrow{C_1}, \overrightarrow{I_1}\}) \text{ s.t. } \{\overrightarrow{C_2}, \overrightarrow{I_2}\}) \ldots) \text{ s.t. } \{\overrightarrow{C_n}, \overrightarrow{I_n}\}) \tag{22a}$$

$$\Rightarrow (p_n \text{ s.t. } \{\overrightarrow{c_n}, \overrightarrow{I_1}\}) \text{ where } \{\, p_n := (p_{n-1} \text{ s.t. } \{\overrightarrow{c_{n-1}}, \overrightarrow{I_{n-1}}\}), \tag{22b}$$

$$\ldots,$$

$$p_3 := (p_2 \text{ s.t. } \{\overrightarrow{c_2}, \overrightarrow{I_2}\}), \tag{22c}$$

$$p_2 := (p_1 \text{ s.t. } \{\overrightarrow{c_1}, \overrightarrow{I_1}\}), \tag{22d}$$

$$\overrightarrow{c_1} := \overrightarrow{C_1}, \; \ldots, \overrightarrow{c_n} := \overrightarrow{C_n} \,\} \tag{22e}$$

Proof.

by induction on $n \geq 1$ and using the reduction rules □

Definition of the Canonical Forms of Restricted Terms: CF5a

$$A \equiv (A_0 \text{ such that } \{ A_1, \ldots, A_n, \overrightarrow{I} \}) \tag{23}$$

- A_i $(i = 1, \ldots, n,\ n \geq 0)$ are proper terms
- $\overrightarrow{I}$ (if not empty) are immediate
- $p_i \in \text{RecV}$ $(i = 1, \ldots, n)$ are fresh

and, for every $i = 0, \ldots, n$:

$$\text{cf}(A_i) \equiv A_{i,0} \text{ where } \{ \overrightarrow{p_i} := \overrightarrow{A_i} \} \quad (k_i \geq 0) \tag{24}$$

(CF5a) If $A_{0,0}$ is immediate, then cf(A) is

$$\text{cf}(A) :\equiv (A_{0,0} \text{ such that } \{ p_1, \ldots, p_n, \overrightarrow{I} \}) \text{ where } \{ \tag{25a}$$

$$\overrightarrow{p_0} := \overrightarrow{A_0}, \tag{25b}$$

$$p_1 := A_{1,0},\ \overrightarrow{p_1} := \overrightarrow{A_1},$$

$$\vdots \tag{25c}$$

$$p_n := A_{n,0},\ \overrightarrow{p_n} := \overrightarrow{A_n} \}$$

Definition of the Canonical Forms of Restricted Terms: CF5b

$$A \equiv (A_0 \text{ such that } \{ A_1, \ldots, A_n, \overrightarrow{I} \}) \tag{26}$$

- A_i $(i = 1, \ldots, n,\ n \geq 0)$ are proper terms
- $\overrightarrow{I}$ (if not empty) are immediate
- $p_i \in \text{RecV}$ $(i = 0, \ldots, n)$ are fresh

and, for every $i = 0, \ldots, n$:

$$\text{cf}(A_i) \equiv A_{i,0} \text{ where } \{ \overrightarrow{p_i} := \overrightarrow{A_i} \} \quad (k_i \geq 0) \tag{27}$$

(CF5b) If $A_{0,0}$ is proper, then cf(A) is:

$$\text{cf}(A) :\equiv (p_0 \text{ such that } \{ p_1, \ldots, p_n, \overrightarrow{I} \}) \text{ where } \{ \tag{28a}$$

$$p_0 := A_{0,0},\ \overrightarrow{p_0} := \overrightarrow{A_0}, \tag{28b}$$

$$p_1 := A_{1,0},\ \overrightarrow{p_1} := \overrightarrow{A_1},$$

$$\vdots \tag{28c}$$

$$p_n := A_{n,0},\ \overrightarrow{p_n} := \overrightarrow{A_n} \}$$

Assume: Terms $=$ Terms$(\mathrm{L}^{\lambda}_{\mathrm{ar}})$, respectively Terms $=$ Terms$(\mathrm{L}^{\lambda}_{\mathrm{ra}})$.

Theorem (Canonical Form Theorem)

For each $A \in$ Terms, there is a unique up to congruence, irreducible term cf$(A) \in$ Terms, such that:

1. *for some explicit, irreducible terms $A_0, \ldots, A_n \in$ Terms $(n \geq 0)$*

$$\mathrm{cf}(A) \equiv A_0 \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\} \tag{29}$$

2. $A \Rightarrow \mathrm{cf}(A)$

Algorithmic Semantic of $\mathrm{L}^{\lambda}_{\mathrm{ar}}$, $\mathrm{L}^{\lambda}_{\mathrm{ra}}$ $(\mathrm{L}^{\lambda}_{\mathrm{r}})$:

- For each proper (i.e, non-immediate) $A \in$ Terms, cf(A) determines the algorithm alg(A) for computing den(A)
- How is the algorithmic semantics of a proper (non-immediate) $A \in$ Terms determined?

Theorem (Effective Reduction Calculi)

For every term $A \in$ Terms, its canonical form cf(A) is effectively computed, by the reduction calculus.

Corollary

Assume the special case of a restrictor term $A \in$ Terms, Terms $=$ Terms$(\mathrm{L}^{\lambda}_{\mathrm{ra}})$*:*

$$A \equiv \big(A_0 \text{ such that } \{ \overrightarrow{C}, \overrightarrow{I} \}\big) \tag{30}$$

- *each term in* $\overrightarrow{I}$ *and in* $\overrightarrow{C}$ *has a type of a truth value*
- *each term in* $\overrightarrow{I}$ *is immediate*
- *each term* C_j *(*$j = 1, \ldots, m$, $m \geq 0$*) in* $\overrightarrow{C}$ *is proper*

Then cf(A) *has the form* (31)*:*

$$\text{cf}(A) \equiv \big(A'_0 \text{ such that } \{ \overrightarrow{c}, \overrightarrow{I} \}\big) \text{ where } \{p_1 := A_1, \ldots, p_n := A_n\} \tag{31}$$

for some immediate $A'_0 \in$ Terms, *some explicit, irreducible* $A_1, \ldots, A_n \in$ Terms *(*$n \geq 0$*), and memory variables* $c_j, p_i \in$ RecV *(*$j = 1, \ldots, m$, $m \geq 0$, $i = 1, \ldots, n$*), such that* $\overrightarrow{c} \subseteq \overrightarrow{p}$*, i.e., for all* j*,*

$$c_j \in \{ p_1, \ldots, p_n \} \tag{32}$$

$$\Phi \equiv \text{The cube is large} \xrightarrow{\text{render}} ? \tag{33}$$

- First Order Logic (FOL) A (available in $\mathrm{L}_{\mathrm{ar}}^{\lambda}$ too)

$$\Phi \xrightarrow{\text{render}} A \equiv \exists x \big[\underbrace{\forall y (cube(y) \leftrightarrow x = y)}_{uniqueness} \wedge isLarge(x) \big] \tag{34}$$

In FOL, by A in (34):

- Existential quantification as the direct, topmost predication
- Uniqueness of the existing entity
- There is no referential force to the object denoted by the NP:

$$[\text{the cube}]_{\mathrm{NP}} \tag{35}$$

- There is no compositional analysis, i.e., no "derivation" of A from the components

- Higher Order Logic (HOL): Henkin (1950) and Mostowski (1957)
  Russellian "the" as a generalized quantifier: lost referential force

$$\text{the} \xrightarrow{\text{render}} [\lambda P \lambda Q [\exists x [\underbrace{\forall y (cube(y) \leftrightarrow x = y)}_{uniqueness} \wedge Q(x)]]] \tag{36a}$$

$$\text{the cube} \xrightarrow{\text{render}} \Big[\lambda P \lambda Q [\exists x [\forall y (cube(y) \leftrightarrow x = y) \wedge Q(x)]]\Big](cube) \tag{36b}$$

(by β reduction)

$$\models\!\!\!\mid\; \lambda Q [\exists x [\forall y (cube(y) \leftrightarrow x = y) \wedge Q(x)]] \tag{36c}$$

$$\Phi \equiv \text{The cube is large} \tag{37a}$$

$$\Phi \xrightarrow{\text{render}} \Big[\lambda Q [\exists x [\forall y (cube(y) \leftrightarrow x = y) \wedge Q(x)]]\Big](isLarge) \tag{37b}$$

(by β reduction)

$$\models\!\!\!\mid\; \exists x [\underbrace{\forall y (cube(y) \leftrightarrow x = y)}_{uniqueness} \wedge Q(x)] \tag{37c}$$

Example: rendering of the definite article "the" Option 1

We may consider rendering the definite article "the" to a constant:

$$\text{the} \xrightarrow{\text{render}} the \in \mathsf{Consts}_{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to \widetilde{\mathsf{e}})} \tag{38}$$

and the following denotation of the constant the:

$$\big[\big(\mathsf{den}(the)\big)(g)\big](\bar{p})(s_0) = \begin{cases} y, & \text{if } y \text{ is the unique } y \in \mathbb{T}_{\mathsf{e}}, \\ & \quad \text{for which } \bar{p}(s \mapsto y)(s_0) = 1 \\ \mathsf{er}, & \text{otherwise} \\ & \quad \text{i.e., there is no unique entity} \\ & \quad \text{that has the property } \bar{p} \text{ in } s_0 \end{cases} \tag{39}$$

for every $\bar{p} \in \mathbb{T}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$ and every $s_0 \in \mathbb{T}_{\mathsf{s}}$

There are other possibilities for rendering the definite article "the", e.g., with complex terms of generalized quantifiers or by using the restrictor.

Example: a constant $unique_0$ for uniqueness of y satisfying a property p in a state s_0

$$unique_0 \in \mathsf{Consts}_{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to (\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}))} \tag{40}$$

For every $\bar{p} \in \mathbb{T}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$, $\bar{q} \in \mathbb{T}_{\widetilde{\mathsf{e}}}$, and every $s_0 \in \mathbb{T}_{\mathsf{s}}$, we can define:

$$\big[\!\!\big(\mathsf{den}(unique_0)\big)\!\!\big](\bar{p})(\bar{q})(s_0) = \begin{cases} 1, & \bar{q}(s_0) \text{ is the unique } y \in \mathbb{T}_{\mathsf{e}} \\ & \text{s.t. } \bar{p}(s \mapsto y)(s_0) = 1 \\ er, & \text{otherwise} \end{cases} \tag{41}$$

- (42a)–(42b) are possible, for some $\bar{p}_0 \in \mathbb{T}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$, $\bar{q}_0 \in \mathbb{T}_{\widetilde{\mathsf{e}}}$, $s_0 \in \mathbb{T}_{\mathsf{s}}$:

$$\begin{aligned} &\bar{q}_0(s_0) = er \text{ and} \\ &\bar{q}_0(s_0) \text{ is the unique } y \in \mathbb{T}_{\mathsf{e}} \text{ s.t. } \bar{p}_0(s \mapsto y)(s_0) = 1 \end{aligned} \tag{42a}$$

$$\big[\!\!\big(\mathsf{den}(unique_0)\big)\!\!\big](\bar{p}_0)(\bar{q}_0)(s_0) = 1 \tag{42b}$$

Example: a constant $unique_1$ for uniqueness of $y \neq er$ satisfying a property p in a state s_0

For every $\bar{p} \in \mathbb{T}_{(\tilde{\mathsf{e}} \to \tilde{\mathsf{t}})}$, $\bar{q} \in \mathbb{T}_{\tilde{\mathsf{e}}}$, $s_0 \in \mathbb{T}_{\mathsf{s}}$,

$$\big[\!\big(\mathrm{den}(unique_1)\big)\!\big](\bar{p})(\bar{q})(s_0) = \begin{cases} 1, & \text{if } \bar{q}(s_0) \text{ is the unique } y \in \mathbb{T}_{\mathsf{e}} \\ & \text{such that } y \neq \text{er and} \\ & \qquad \bar{p}(s \mapsto y)(s_0) = 1 \\ \text{er}, & \text{otherwise} \end{cases} \tag{43}$$

- It is possible that for some $\bar{p}_0 \in \mathbb{T}_{(\tilde{\mathsf{e}} \to \tilde{\mathsf{t}})}$, $\bar{q}_0 \in \mathbb{T}_{\tilde{\mathsf{e}}}$, $s_0 \in \mathbb{T}_{\mathsf{s}}$:

$$\text{for all } x\Big[[x \neq er \;\&\; \bar{p}_0(s \mapsto x)(s_0) = 1] \iff x = \bar{q}_0(s_0)\Big] \tag{44a}$$

$$\bar{p}_0(s \mapsto er)(s_0) = 1 \tag{44b}$$

$\therefore$ Both $\bar{q}_0(s_0) \neq er$ and er have the property $\bar{p}_0$ in s_0, i.e.,
$\bar{q}_0(s_0) \neq er$ is not per se unique entity having the property $\bar{p}_0$ in s_0

Example: a constant *unique* for uniqueness of y satisfying a property p in a state s_0

For every $\bar{p} \in \mathbb{T}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$, $\bar{q} \in \mathbb{T}_{\widetilde{\mathsf{e}}}$, $s_0 \in \mathbb{T}_{\mathsf{s}}$,

$$\big[\big(\mathsf{den}(unique)\big)\big](\bar{p})(\bar{q})(s_0) = \begin{cases} 1, & \text{if } \bar{q}(s_0) \neq er \text{ and} \\ & \bar{q}(s_0) \text{ is the unique } y \in \mathbb{T}_{\mathsf{e}} \\ & \text{such that } \bar{p}(s \mapsto y)(s_0) = 1 \\ \text{er}, & \text{otherwise} \end{cases} \tag{45}$$

Therefore:

$$\text{exists } y \; \big[y = \bar{q}(s_0) \neq er \;\; \& \;\; \text{for all } x \; [\bar{p}(s \mapsto x)(s_0) = 1 \iff x = y]\big] \tag{46a}$$

$$\bar{q}(s_0) \neq er \;\; \& \;\; \text{for all } x \; [\bar{p}(s \mapsto x)(s_0) = 1 \iff x = \bar{q}(s_0)] \tag{46b}$$

Example: possible rendering of the determiner "the" and definite descriptors — Option 3

We can render the definite article "the" to A_1 that is underspecified for p:

$$\text{the} \xrightarrow{\text{render}} A_1 \equiv \big(q \text{ s.t. } \{\, unique(p)(q) \,\}\big) : \widetilde{\mathsf{e}} \tag{47a}$$

$$p \in \mathsf{RecV}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}, \quad q \in \mathsf{RecV}_{\widetilde{\mathsf{e}}} \tag{47b}$$

- Then, p gets specified, by nouns in NPs:

$$\text{the cube} \xrightarrow{\text{render}} A_2 : \widetilde{\mathsf{e}} \tag{48a}$$

$$A_2 \equiv \big(q \text{ s.t. } \{\, unique(p)(q) \,\}\big) \text{ where } \{\, p := \mathit{cube} \,\} \tag{48b}$$

by (st1)

$$\Rightarrow_{\mathsf{cf}} \big(q \text{ s.t. } \{\, U \,\}\big) \text{ where } \{\, U := unique(p)(q), \; p := \mathit{cube} \,\} \tag{48c}$$

$$\text{The cube is large} \xrightarrow{\text{render}} A_3 : \widetilde{\mathsf{t}} \tag{49a}$$

$$A_3 \equiv isLarge\Big((q \text{ s.t. } \{ unique(p)(q) \}) \text{ where } \{ p := cube \}\Big) \tag{49b}$$

by (ap)

$$\Rightarrow isLarge(Q) \text{ where } \{ Q := \big[(q \text{ s.t. } \{ unique(p)(q) \}) \text{ where } \{ p := cube \}\big] \} \tag{49c}$$

by (wh-comp) from(48c)

$$\Rightarrow_{\mathsf{cf}} isLarge(Q) \text{ where } \{ Q := (q \text{ s.t. } \{ U \}), U := unique(p)(q), p := cube \} \tag{49d}$$

Algorithmic Pattern: definite descriptors in predicative sentences: Opt3

$$A \equiv L(Q) \text{ where } \{ Q := (q \text{ s.t. } \{ U \}), U := unique(p)(q) \} \tag{50a}$$

$$p, q, L \in \mathsf{FreeV}(A),\ p \in \mathsf{RecV}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})},\ q \in \mathsf{RecV}_{\widetilde{\mathsf{e}}}, \tag{50b}$$

$$Q \in \mathsf{RecV}_{\widetilde{\mathsf{e}}},\ U \in \mathsf{RecV}_{\widetilde{\mathsf{t}}}, L \in \mathsf{RecV}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})} \tag{50c}$$

$$\text{The cube } n \text{ is large} \xrightarrow{\text{render}} A_4 : \widetilde{\mathsf{t}} \tag{51a}$$

$$\begin{aligned} A_4 \equiv isLarge\Big(\big(q \text{ s.t. } \{\, unique(N)(q),\ p(q) \,\}\big) \text{ where } \{ \\ q := n,\ p := \mathsf{cube},\ N := named\text{-}n \,\}\Big) \end{aligned} \tag{51b}$$

$$\begin{aligned} \Rightarrow_{\mathsf{cf}} isLarge(Q) \text{ where } \{\, Q := \big(q \text{ s.t. } \{\, U, C \,\}\big), \\ U := unique(N)(q),\ C := p(q), \\ q := n,\ p := \mathsf{cube},\ N := named\text{-}n \,\} \end{aligned} \tag{51c}$$

- direct reference; uniqueness and existence are consequences

$$\text{The cube } n \text{ is large} \xrightarrow{\text{render}} A_5 : \widetilde{\mathsf{t}} \tag{52a}$$

$$\begin{aligned} A_5 \equiv isLarge\Big(\big(q \text{ s.t. } \{\, p(q) \,\}\big) \text{ where } \{ \\ q := n,\ p := \mathsf{cube} \,\}\Big) \end{aligned} \tag{52b}$$

$$\begin{aligned} \Rightarrow_{\mathsf{cf}} isLarge(Q) \text{ where } \{\, Q := \big(q \text{ s.t. } \{\, C \,\}\big),\ C := p(q), \\ q := n,\ p := \mathsf{cube} \,\} \end{aligned} \tag{52c}$$

$$\text{the} \xrightarrow{\text{render}} B_1^{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to \widetilde{\mathsf{e}})}$$

$$\equiv \lambda(x)\Big(\big[q \text{ s.t. } \{\, unique(p)(q) \,\}\big] \text{ where } \{\, p := x \,\}\Big) \tag{53a}$$

by (st1), (wh-comp), (λ-comp)

$$\Rightarrow \lambda(x)\Big(\Big[\big[q \text{ s.t. } \{\, U \,\}\big] \text{ where } \{ U := unique(p)(q) \,\}\Big] \text{ where } \{\, p := x \,\}\Big) \tag{53b}$$

by (head), (λ-comp)

$$\Rightarrow \lambda(x)\Big(\big[q \text{ s.t. } \{\, U \,\}\big] \text{ where } \{ U := unique(p)(q),\ p := x \,\}\Big) \tag{53c}$$

by (λ)

$$\Rightarrow \lambda(x)\big[q \text{ s.t. } \{\, U'(x) \,\}\big] \text{ where } \{ U' := \lambda(x)unique(p'(x))(q),\ p' := \lambda(x)(x) \,\} \tag{53d}$$

$$\text{the cube} \xrightarrow{\text{render}} B_2 : \widetilde{\mathsf{e}} \qquad \text{by (53d) and functional application} \tag{54a}$$

$$\begin{aligned} B_2 \equiv \Big[\lambda(x)\big[q \text{ s.t. } \{\, U'(x) \,\}\big] \text{ where } \{ \\ U' &:= \lambda(x)unique(p'(x))(q), \\ p' &:= \lambda(x)(x) \,\}\Big](cube) \end{aligned} \tag{54b}$$

by (recap)

$$\begin{aligned} \Rightarrow \Big[\lambda(x)[q \text{ s.t. } \{\, U'(x) \,\}]\Big](cube) \text{ where } \{ \\ U' &:= \lambda(x)unique(p'(x))(q), \\ p' &:= \lambda(x)(x) \,\} \end{aligned} \tag{54c}$$

by (ap)

$$\begin{aligned} \Rightarrow \Big[\big[\lambda(x)[q \text{ s.t. } \{\, U'(x) \,\}]\big](c) \text{ where } \{\, c := cube \,\}\Big] \\ \text{where } \{\, U' &:= \lambda(x)unique(p'(x))(q), \\ p' &:= \lambda(x)(x) \,\} \end{aligned} \tag{54d}$$

by (head)

$$\begin{aligned} \Rightarrow \Big[\big[\lambda(x)[q \text{ s.t. } \{\, U'(x) \,\}]\big](c)\Big] \\ \text{where } \{\, U' &:= \lambda(x)unique(p'(x))(q), \\ p' &:= \lambda(x)(x), \ c := cube \,\} \end{aligned} \tag{54e}$$

$$\text{The cube is large} \xrightarrow{\text{render}} B_3 : \widetilde{\mathsf{t}} \tag{55a}$$

$$\begin{aligned} B_3 \equiv isLarge\Big(\Big[\big[\lambda(x)[q \text{ s.t. } \{\, U'(x) \,\}]\big](c)\Big] \\ \text{where } \{\, U' := \lambda(x)unique(p'(x))(q), \\ p' := \lambda(x)(x),\ c := cube \,\}\Big) \end{aligned} \tag{55b}$$

by (ap)

$$\begin{aligned} \Rightarrow isLarge(Q) \text{ where } \{ \\ Q := \Big(\Big[\big[\lambda(x)[q \text{ s.t. } \{\, U'(x) \,\}]\big](c)\Big] \\ \text{where } \{\, U' := \lambda(x)unique(p'(x))(q), \\ p' := \lambda(x)(x),\ c := cube \,\}\Big) \,\} \end{aligned} \tag{55c}$$

by (B-S)

$$\begin{aligned} \Rightarrow isLarge(Q) \text{ where } \{\, Q := \big[\lambda(x)[q \text{ s.t. } \{\, U'(x) \,\}]\big](c), \\ U' := \lambda(x)unique(p'(x))(q), \\ p' := \lambda(x)(x),\ c := cube \,\} \end{aligned} \tag{55d}$$

Outlook1: Development of Computational Theories and Applications

- Generalised Computational Grammar: CompSynSem interfaces in HL
  - Hierarchical lexicon with morphological structure and lexical rules
  - Syntax of HL expressions (phrasal and grammatical dependences)
  - Syntax-semantics inter-relations in lexicon and phrases
- A Big Picture — simplified and approximated, but realistic:

Algorithmic CompSynSem of Human Language (HL)

$$\underbrace{\text{HL Syn} \Longleftrightarrow \underbrace{L_{ar}^{\lambda} \,/\, L_{r}^{\lambda} \,/\, SitI \xrightarrow{\text{Reduction Calc}} \text{Canonical Forms}}_{\text{Canonical Computations}} \Longrightarrow \text{Denotations}}_{\text{(Canonically) Algorithmic CompSynSem Interfaces}}$$

(I've done quite a lot of it, but still a lot to do!)

Some Current Tasks (among many others) and Future Work

- My focus is on:
  - Development of L_{ar}^{λ} and L_{r}^{λ}
  - Dependent-Type Theory of Situated Information and Algorithms
  - Applications to formal and natural languages
    - Extending the Coverage of Computational Semantics
    - Computational Syntax-Semantics Interfaces
    - Semantics of programming and specification languages
    - Theoretical foundations of (parts of) compilers
- More to come

THANK YOU!

Some References I

Roussanka Loukanova.
Type-Theory of Parametric Algorithms with Restricted Computations.
In *Distributed Computing and Artificial Intelligence, 17th International Conference*, pages 321–331, Cham, 07 August 2020 2021. Springer International Publishing.
URL: https://doi.org/10.1007/978-3-030-53036-5_35.

Yiannis N Moschovakis.
The formal language of recursion.
Journal of Symbolic Logic, 54(04):1216–1252, 1989.
URL: https://doi.org/10.1017/S0022481200041086.

Yiannis N. Moschovakis.
A Logical Calculus of Meaning and Synonymy.
Linguistics and Philosophy, 29(1):27–89, 2006.
URL: https://doi.org/10.1007/s10988-005-6920-7.