Restricted Quantification in New Type-Theory of Algorithms

Roussanka Loukanova

Seminar of Algebra and Logic (seminarAiL)
Department of Algebra and Logic
Institute of Mathematics and Informatics (IMI)
Bulgarian Academy of Sciences (BAS), Bulgaria

Fri Dec 3 CET 2021

Development of Moschovakis Type-Theory of Algorithms: a new math of algorithms

- Moschovakis [7], 2006, Type-Theory of Acyclic Recursion $\mathrm{L}^{\lambda}_{\mathrm{ar}}$ computations, by saving the algorithmic steps in memory locations
- Loukanova 2021, 2022: extended work
  - extended reduction calculi in $\mathrm{L}^{\lambda}_{\mathrm{ar}}$, e.g.:
    chain reduction; logic operators, quantifiers
  - mathematics of parametric / underspecified algorithms
    dependent on restricted recursion variables, which can be underspecified or instantiated by specification
  - restrictor operator Loukanova [6] — constrained computations
    differently before in SitT & new since 2020 (DTTSitI extending SitT)
  - representing scope ambiguity in natural language, incl. in math text, via multiple quantifiers in $\mathrm{L}^{\lambda}_{\mathrm{ar}}$-terms, extended with constraints
  - existential quantification in $\mathrm{L}^{\lambda}_{\mathrm{ar}}$: introduction and elimination rules
    (extending: Loukanova 2015)
- logic operators and quantification in $\mathrm{L}^{\lambda}_{\mathrm{ar}}$ (new), in this talk
- existential quantification in $\mathrm{L}^{\lambda}_{\mathrm{ar}}$: introduction and elimination rules (new)
- universal quantification in $\mathrm{L}^{\lambda}_{\mathrm{ar}}$: introduction and elimination rules (new)

Gallin Types: $\sigma :\equiv \mathsf{e} \mid \mathsf{t} \mid \mathsf{s} \mid (\tau_1 \to \tau_2)$ (Gallin, 1975)

For all $\tau \in$ Types:

Constants: $\mathsf{Consts}_\tau = \{\mathsf{c}^\tau_0, \mathsf{c}^\tau_1, \ldots, \mathsf{c}^\tau_{k_\tau}\}$

Variables: $\mathsf{PureV}_\tau = \{v^\tau_0, v^\tau_1, \ldots\}$

$\mathsf{MemoryV}_\tau = \mathsf{RecV}_\tau = \{p^\tau_0, p^\tau_1, \ldots\}$

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

$$\mathsf{A} :\equiv \mathsf{c}^\tau : \tau \mid x^\tau : \tau \quad (\text{for } \mathsf{c}^\tau \in \mathsf{Consts}_\tau,\ x^\tau \in \mathsf{PureV}_\tau \cup \mathsf{RecV}_\tau) \tag{1a}$$

$$\mid \mathsf{B}^{(\sigma\to\tau)}(\mathsf{C}^\sigma) : \tau \tag{1b}$$

$$\mid \lambda(v^\sigma)\,(\mathsf{B}^\tau) : (\sigma \to \tau) \quad (\text{for } v^\sigma \in \mathsf{PureV}_\sigma) \tag{1c}$$

$$\mid \left[\mathsf{A}^{\sigma_0}_0 \text{ where } \{\, p^{\sigma_1}_1 := \mathsf{A}^{\sigma_1}_1, \ldots, p^{\sigma_i}_i := \mathsf{A}^{\sigma_i}_i, \ldots, p^{\sigma_n}_n := \mathsf{A}^{\sigma_n}_n \,\}\right] : \sigma_0 \tag{1d}$$

$$\mid \left[\mathsf{A}^{\sigma_0}_0 \text{ such that } \{\, \mathsf{C}^{\tau_1}_1, \ldots, \mathsf{C}^{\tau_m}_m \,\}\right] : \sigma'_0 \tag{1e}$$

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

$$some, every \in \mathsf{Consts}_{((\widetilde{\mathsf{e}}\to\widetilde{\mathsf{t}})\to((\widetilde{\mathsf{e}}\to\widetilde{\mathsf{t}})\to\widetilde{\mathsf{t}}))} \tag{2}$$

$$cat, sleeps, number \in \mathsf{Consts}_{(\widetilde{\mathsf{e}}\to\widetilde{\mathsf{t}})}, \quad \geq \in \mathsf{Consts}_{(\widetilde{\mathsf{e}}\to(\widetilde{\mathsf{e}}\to\widetilde{\mathsf{t}}))},$$

$$\lambda x(cat(x)) \approx cat, \quad \lambda x(sleeps(x)) \approx sleeps \tag{3}$$

$$\text{Some cat sleeps.} \xrightarrow[(opt1)]{\text{render}} some(cat)(sleeps) \tag{4a}$$

$$\Rightarrow_{\mathsf{cf}} some(p_1)(p_2) \text{ where } \{p_1 := cat, \; p_2 := sleeps\} \tag{4b}$$

$$(\text{by (3)}) \quad \approx some(p_1)(p_2) \text{ where } \{p_1 := \lambda x(cat(x)), \; p_2 := \lambda x(sleeps(x))\} \tag{4c}$$

$$\text{Some cat sleeps.} \xrightarrow[(opt2)]{\text{render}} E \text{ where } \{E := \underbrace{some(p_1)}_{Q}(p_2), \; p_1 := cat, \; p_2 := sleeps\} \tag{5a}$$

Generalised Binary Quantifiers: Rendering Options 1,2,3

Some cat sleeps. $\xrightarrow[(opt1)]{\text{render}}$ $some(cat)(sleeps)$ (6a)

$\Rightarrow_{\text{cf}}$ $\underbrace{some(p_1)}_{Q}(p_2)$ where $\{p_1 := cat,\ p_2 := sleeps\}$ (6b)

Some cat sleeps. $\xrightarrow[(opt2)]{\text{render}}$ E where $\{E := \underbrace{some(p_1)}_{Q}(p_2),$ (7a)

$p_1 := cat,\ p_2 := sleeps\}$ (7b)

Some cat sleeps. $\xrightarrow[(opt3)]{\text{render}}$ E where $\{E := Q(p_2),$ (8a)

$Q := q(p_1),\ q := some,$ (8b)

$p_1 := cat,\ p_2 := sleeps\}$ (8c)

Existential Instantiation R, cf. [2] (ExIR)

For $d \in \mathsf{Im}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$ (state-dependent property),
$q \in \mathsf{RecV}_{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to ((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to \widetilde{\mathsf{t}}))}$ (binary quantifier),
$Q \in \mathsf{RecV}_{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to \widetilde{\mathsf{t}})}$ (one-argument quantifier),
$A \in \mathsf{Terms}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$ (state-dependent property), such that A is immediate,
and any fresh recursion variable $c \in \mathsf{RecV}_{\widetilde{\mathsf{e}}}$

$$
\begin{aligned}
&\big[E \text{ where } \{\, E := Q(A), && \text{(9a)}\\
&\qquad\qquad Q := q(d),\ q := \mathit{some}, && \text{(9b)}\\
&\qquad\qquad \overrightarrow{d} := \overrightarrow{A}\,\}\big] && \text{(9c)}\\
&\quad \text{s.t. } \{\, \overrightarrow{C}\,\} && \text{(9d)}\\
\vdash\ &\\
&\big[R' \text{ where } \{\, R' := A(c), && \text{(9e)}\\
&\qquad\qquad \overrightarrow{d} := \overrightarrow{A}\,\}\big] && \text{(9f)}\\
&\quad \text{s.t. } \{\, d(c),\ \overrightarrow{C}\,\} && \text{(9g)}
\end{aligned}
$$

d can be instantiated at a later stage, or is already instantiated:

$$d := D \in \overrightarrow{d} := \overrightarrow{A} \tag{10}$$

Existential Instantiation Head1 (ExIH1)

For $d \in \mathsf{Im}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$ (state-dependent property),
$q \in \mathsf{RecV}_{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to ((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to \widetilde{\mathsf{t}}))}$ (binary quantifier),
$Q \in \mathsf{RecV}_{((\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}}) \to \widetilde{\mathsf{t}})}$ (one-argument quantifier),
$A \in \mathsf{Terms}_{(\widetilde{\mathsf{e}} \to \widetilde{\mathsf{t}})}$ (state-dependent property), such that A is immediate,
and any fresh recursion variable $c \in \mathsf{RecV}_{\widetilde{\mathsf{e}}}$
(ExIH1a)

$$[Q(A) \text{ where } \{ Q := q(d),\ q := some,\ \overrightarrow{a} := \overrightarrow{A} \}] \tag{11a}$$

$$\text{s.t. } \{ \overrightarrow{C} \} \tag{11b}$$

$$\vdash [A(c) \text{ where } \{ \overrightarrow{a} := \overrightarrow{A} \}] \text{ s.t. } \{ d(c), \overrightarrow{C} \} \tag{11c}$$

(ExIH1b)

$$\left[some(d)(A) \text{ where } \{ \overrightarrow{a} := \overrightarrow{A} \}\right] \text{ s.t. } \{ \overrightarrow{C} \} \tag{12a}$$

$$\vdash \left[A(c) \text{ where } \{ \overrightarrow{a} := \overrightarrow{A} \}\right] \text{ s.t. } \{ d(c), \overrightarrow{C} \} \tag{12b}$$

d can be instantiated at a later stage, or is already instantiated:

$$d := D \in \overrightarrow{a} := \overrightarrow{A} \tag{13}$$

Existential Introduction Rule R1, cf. [2]

(ExistIntro)

For any $c \in \mathsf{RecV}_{\widetilde{\mathsf{e}}}$

$$R' \text{ where } \{\, R' := A\{\, x :\equiv c \,\},\ d := D, \tag{14a}$$
$$\overrightarrow{d} := \overrightarrow{A}, \tag{14b}$$
$$c := C \quad \text{(optional)} \,\} \tag{14c}$$
$$\text{s.t. } \{\, d(c),\ \overrightarrow{C} \,\} \tag{14d}$$
$$\vdash$$
$$E \text{ where } \{\, E := Q(R), \tag{14e}$$
$$R := \lambda(x)A,\ Q := q(d),\ q := some,\ d := D, \tag{14f}$$
$$\overrightarrow{d} := \overrightarrow{A} \,\} \tag{14g}$$
$$\text{s.t. } \{\, \overrightarrow{C} \,\} \tag{14h}$$

There are other alternative rules for the existential introduction.

Scope Ambiguity: *de dicto* / de re rendering of quantifiers, cf. [1, 5]

Every natural number is larger than some number (15a)

Every natural number x_2 is larger than some number x_1 (15b)

Every professor reads some paper (de dicto)

$$every(professor)(\lambda(x_2)some(paper)(\lambda(x_1)read(x_1)(x_2))) \quad (16a)$$

$$\Rightarrow_{\mathsf{cf}} every(p)(R_2) \text{ where } \{R_2 := \lambda(x_2)some(b'(x_2))(R_1(x_2)), \quad (16b)$$

$$R_1 := \lambda(x_2)\,\lambda(x_1)read(x_1)(x_2), \quad (16c)$$

$$p := professor, \quad (16d)$$

$$b' := \lambda(x_2)paper\} \quad (16e)$$

$$\Rightarrow_{\mathsf{gcf}} every(p)(R_2) \text{ where } \{R_2 := \lambda(x_2)some(b)(R_1(x_2)), \quad (16f)$$

$$R_1 := \lambda(x_2)\,\lambda(x_1)read(x_1)(x_2), \quad (16g)$$

$$p := professor,\ b := paper\} \quad (16h)$$

Scope Ambiguity: *de re* / de dicto rendering of quantifiers

Every professor reads some paper $\xrightarrow{\text{render}}$ (de re)

$$some(paper)(\lambda(x_1)every(professor)(\lambda(x_2)read(x_1)(x_2))) \quad (17a)$$

$\Rightarrow_{\text{cf}}$ $some(b)(R_1)$ where $\{R_1 := \lambda(x_1)every(p'(x_1))(R_2(x_1)),$ (17b)

$R_2 := \lambda(x_1)\,\lambda(x_2)read(x_1)(x_2),$ (17c)

$p' := \lambda(x_1)professor,$ (17d)

$b := paper\}$ (17e)

$\Rightarrow_{\text{gcf}}$ $some(b)(R_1)$ where $\{R_1 := \lambda(x_1)every(p)(R_2(x_1)),$ (17f)

$R_2 := \lambda(x_1)\,\lambda(x_2)read(x_1)(x_2),$ (17g)

$p := professor,\ b := paper\}$ (17h)

$\approx$ $some(b)(R_1)$ where $\{R_1 := \lambda(x_1)every(p)(R_2(x_1)),$ (17i)

$R_2 := read,$ (17j)

$p := professor,\ b := paper\}$ (17k)

de dicto and *de re* renderings of quantifiers: shared algorithmic pattern

Every professor reads some paper $\xrightarrow{\text{render}}$ (de dicto)

$$R_3 \text{ where } \{R_3 := every(p)(R_2), \tag{18a}$$
$$R_2 := \lambda(x_2)some(b)(R_1(x_2)), \tag{18b}$$
$$R_1 := \lambda(x_2)\,\lambda(x_1)read(x_1)(x_2), \tag{18c}$$
$$p := professor,\ b := paper\} \tag{18d}$$

Every professor reads some paper $\xrightarrow{\text{render}}$ (de re)

$$R_3 \text{ where } \{R_3 := some(b)(R_1), \tag{19a}$$
$$R_1 := \lambda(x_1)every(p)(R_2(x_1)), \tag{19b}$$
$$R_2 := \lambda(x_1)\,\lambda(x_2)read(x_1)(x_2), \tag{19c}$$
$$p := professor,\ b := paper\} \tag{19d}$$

specified algorithm: de dicto permmutation of the quantifiers

$$
\begin{aligned}
S_{21} \equiv R_3 \text{ where } \{\, R_3 &:= Q_2(R_2), && \text{(20a)}\\
R_2 &:= \lambda(x_2)Q_1(R_1^1(x_2)), && \text{(20b)}\\
R_1^1 &:= \lambda(x_2)\,\lambda(x_1)h(x_1)(x_2), && \text{(20c)}\\
Q_1 &:= q_1(d_1),\ Q_2 := q_2(d_2), && \text{(20d)}\\
q_1 &:= \textit{some},\ d_1 := \textit{paper}, && \text{(20e)}\\
q_2 &:= \textit{every},\ d_2 := \textit{professor},\ h := \textit{read}\,\} && \text{(20f)}
\end{aligned}
$$

specified algorithm: de re permmutation of the quantifiers

$$
\begin{aligned}
S_{12} \equiv R_3 \text{ where } \{\, R_3 &:= Q_1(R_1), && \text{(21a)}\\
R_1 &:= \lambda(x_1)Q_2(R_2^1(x_1)), && \text{(21b)}\\
R_2^1 &:= \lambda(x_1)\,\lambda(x_2)h(x_1)(x_2), && \text{(21c)}\\
Q_1 &:= q_1(d_1),\ Q_2 := q_2(d_2), && \text{(21d)}\\
q_1 &:= \textit{some},\ d_1 := \textit{paper}, && \text{(21e)}\\
q_2 &:= \textit{every},\ d_2 := \textit{professor},\ h := \textit{read}\,\} && \text{(21f)}
\end{aligned}
$$

- Underspecified algorithms for d_i, $i \in \{1, 2\}$
- Permmutation of the order of binding of the arguments of h via recursive quantifier binding
  - Specified de dicto: fixed permmutation of the quantifiers

$$
\begin{aligned}
S_{21} \equiv R_3 \text{ where } \{\, R_3 &:= Q_2(R_2), && \text{(22a)}\\
R_2 &:= \lambda(x_2) Q_1(R_1^1(x_2)), && \text{(22b)}\\
R_1^1 &:= \lambda(x_2)\, \lambda(x_1) h(x_1)(x_2), && \text{(22c)}\\
Q_1 &:= q_1(d_1),\ Q_2 := q_2(d_2), && \text{(22d)}\\
q_1 &:= \textit{some},\ q_2 := \textit{every}\, \} && \text{(22e)}
\end{aligned}
$$

  - Specified de re: fixed permmutation of the quantifiers

$$
\begin{aligned}
S_{12} \equiv R_3 \text{ where } \{\, R_3 &:= Q_1(R_1), && \text{(23a)}\\
R_1 &:= \lambda(x_1) Q_2(R_2^1(x_1)), && \text{(23b)}\\
R_2^1 &:= \lambda(x_1)\, \lambda(x_2) h(x_1)(x_2), && \text{(23c)}\\
Q_1 &:= q_1(d_1),\ Q_2 := q_2(d_2), && \text{(23d)}\\
q_1 &:= \textit{some},\ q_2 := \textit{every}\, \} && \text{(23e)}
\end{aligned}
$$

Dynamic query-answers in underspecified database via Existential Instantiation, cf. [2]

Underspecified permutation of the order of binding of the arguments of h

$$U \equiv R_3 \text{ where } \{\, l_1 := Q_1(R_1),\ l_2 := Q_2(R_2), \tag{24a}$$
$$Q_1 := q_1(d_1),\ Q_2 := q_2(d_2), \tag{24b}$$
$$q_1 := \mathit{some},\ q_2 := \mathit{every}, \tag{24c}$$
$$h := \mathit{read},\ d_1 := \mathit{paper}, \tag{24d}$$
$$d_2 := \mathit{professor}\,\} \tag{24e}$$
$$\text{s.t. } \{\, R_3 \text{ recursively binds to each } Q_i \text{ (for } i = 1, 2), \tag{24f}$$
$$Q_i \text{ binds the } i\text{-th argument of } h\,\} \tag{24g}$$

$$R_3 \text{ specified-to } ? \tag{24h}$$

$$\text{Which is this paper?} \xrightarrow{\text{render}} \tag{25a}$$

$$R_3 \text{ where } \{\, R_3 := Q_1(R_1), \tag{25b}$$

$$R_1 := \lambda(x_1) Q_2(R_2^1(x_1)), \tag{25c}$$

$$R_2^1 := \lambda(x_1)\, \lambda(x_2) h(x_1)(x_2), \tag{25d}$$

$$Q_1 := q_1(d_1),\ Q_2 := q_2(d_2), \tag{25e}$$

$$q_2 := \mathit{every},\ d_2 := \mathit{professor}, \tag{25f}$$

$$h := \mathit{read},\ q_1 := \mathit{some},\ d_1 := \mathit{paper}\,\} \tag{25g}$$

$$\vdash \text{ by existential instantiation} \tag{26a}$$

$$R_1' \text{ where } \{\, R_1' := Q_2(R_2^1(c)), \tag{26b}$$

$$R_2^1 := \lambda(x_1)\, \lambda(x_2) h(x_1)(x_2), \tag{26c}$$

$$Q_2 := q_2(d_2), \tag{26d}$$

$$q_2 := \mathit{every},\ d_2 := \mathit{professor}, \tag{26e}$$

$$h := \mathit{read},\ d_1 := \mathit{paper}\,\} \tag{26f}$$

$$\text{s.t. } \{\, d_1(c)\,\} \tag{26g}$$

$$c \text{ specified-to?} \tag{26h}$$

- In case there is such a term C, the system responds:

$$\vdash \tag{27a}$$
$$R'_1 \text{ where } \{\, R'_1 := Q_2(R^1_2(c)), \tag{27b}$$
$$R^1_2 := \lambda(x_1)\,\lambda(x_2)h(x_1)(x_2), \tag{27c}$$
$$Q_2 := q_2(d_2), \tag{27d}$$
$$q_2 := \mathit{every},\ d_2 := \mathit{professor}, \tag{27e}$$
$$h := \mathit{read},\ d_1 := \mathit{paper},\ c := C \,\} \tag{27f}$$
$$\text{s.t. } \{\, d_1(c) \,\} \tag{27g}$$
$$c \text{ specified-to } C \tag{27h}$$

- Otherwise, the system answers:

$$\text{There is no such } c. \tag{28a}$$
$$U \text{ specified-to } S_{21} \qquad \text{answer} \tag{28b}$$

Reductiona and Canonical Forms of $\mathrm{L}^{\lambda}_{\mathrm{ar}}$ Are they carried over the rules of quantifiers?

- The original $\mathrm{L}^{\lambda}_{\mathrm{ar}}$: Canonical Form Theorem, Moschovakis [7], 2006
- Reduction Calculi, for a variety of extended $\mathrm{L}^{\lambda}_{\mathrm{ar}}$

Theorem (γ^*-Canonical Form Theorem, Loukanova [4])

For every $A \in$ Terms, *there is a unique up to congruence,* γ^**-irreducible* $\mathsf{cf}_{\gamma^*}(A) \in$ Terms *s.th.:*

1. $\mathsf{cf}_{\gamma^*}(A) \equiv A_0$ where $\{p_1 := A_1, \ldots, p_n := A_n\}$
   for some explicit, γ^**-irreducible* $A_0, \ldots, A_n \in$ Terms *(*$n \geq 0$*)*
2. $A \Rightarrow_{\gamma^*} \mathsf{cf}_{\gamma^*}(A)$

Theorem (γ-Canonical Form Theorem, Loukanova [3])

For every $A \in$ Terms, *there is a unique up to congruence,* γ*-irreducible* $\mathsf{cf}_{\gamma}(A) \in$ Terms *s.th.:*

1. $\mathsf{cf}_{\gamma}(A) \equiv A_0$ where $\{p_1 := A_1, \ldots, p_n := A_n\}$
   for some explicit, γ*-irreducible* $A_0, \ldots, A_n \in$ Terms *(*$n \geq 0$*)*
2. $A \overset{i\gamma}{\Rightarrow}_{\gamma} \mathsf{cf}_{\gamma}(A)$ *(innermost reductions!)*

Some References I

Loukanova, R.: Relationships between Specified and Underspecified Quantification by the Theory of Acyclic Recursion.
ADCAIJ: Advances in Distributed Computing and Artificial Intelligence Journal **5**(4), 19–42 (2016).
URL https://doi.org/10.14201/ADCAIJ2016541942

Loukanova, R.: Specification of Underspecified Quantifiers via Question-Answering by the Theory of Acyclic Recursion.
In: T. Andreasen, H. Christiansen, J. Kacprzyk, H. Larsen, G. Pasi, O. Pivert, G. De Tré, M.A. Vila, A. Yazici, S. Zadrożny (eds.) Flexible Query Answering Systems 2015, *Advances in Intelligent Systems and Computing*, vol. 400, pp. 57–69. Springer International Publishing (2016).
URL https://doi.org/10.1007/978-3-319-26154-6_5

Some References II


Loukanova, R.: Gamma-Reduction in Type Theory of Acyclic Recursion.
Fundamenta Informaticae **170**(4), 367–411 (2019).
URL https://doi.org/10.3233/FI-2019-1867


Loukanova, R.: Gamma-Satar Canonical Forms in the Type-Theory of Acyclic Algorithms.
In: J. van den Herik, A.P. Rocha (eds.) Agents and Artificial Intelligence. ICAART 2018, *Lecture Notes in Computer Science*, vol. 11352, pp. 383–407. Springer International Publishing, Cham (2019).
URL https://doi.org/10.1007/978-3-030-05453-3_18

Some References III


Loukanova, R.: Type-theory of acyclic algorithms for models of consecutive binding of functional neuro-receptors.
In: A. Grabowski, R. Loukanova, C. Schwarzweller (eds.) AI Aspects in Reasoning, Languages, and Computation, vol. 889, pp. 1–48. Springer International Publishing, Cham (2020).
URL https://doi.org/10.1007/978-3-030-41425-2_1


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


Moschovakis, Y.N.: 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