Mathematical Synthesis
of
Equational Deduction Systems

Marcelo Fiore

COMPUTER LABORATORY
UNIVERSITY OF CAMBRIDGE

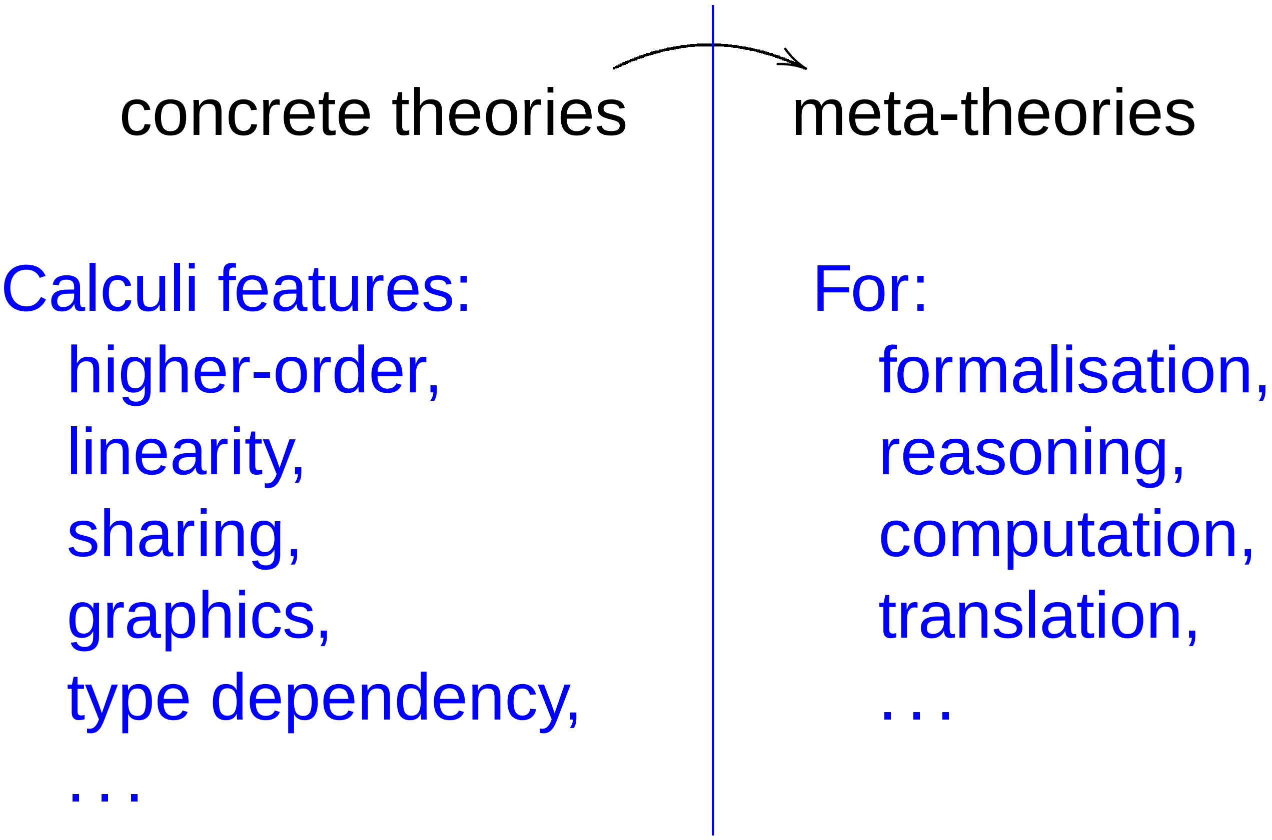
TLCA 2009
3.VII.2009

Context

concrete theories

meta-theories

Context



concrete theories

Calculi features:

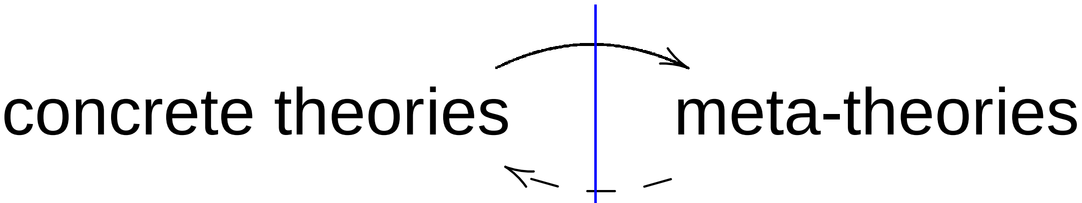
- higher-order,
- linearity,
- sharing,
- graphics,
- type dependency,
- . . .

meta-theories

For:

- formalisation,
- reasoning,
- computation,
- translation,
- . . .

Context

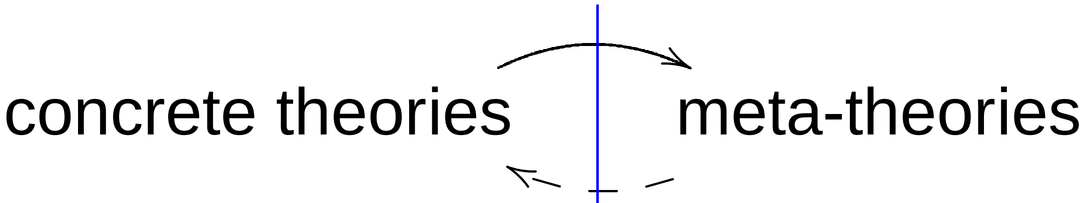

concrete theories

meta-theories

Calculi features:
- higher-order,
- linearity,
- sharing,
- graphics,
- type dependency,
- …

For:
- formalisation,
- reasoning,
- computation,
- translation,
- …

Context


concrete theories

meta-theories

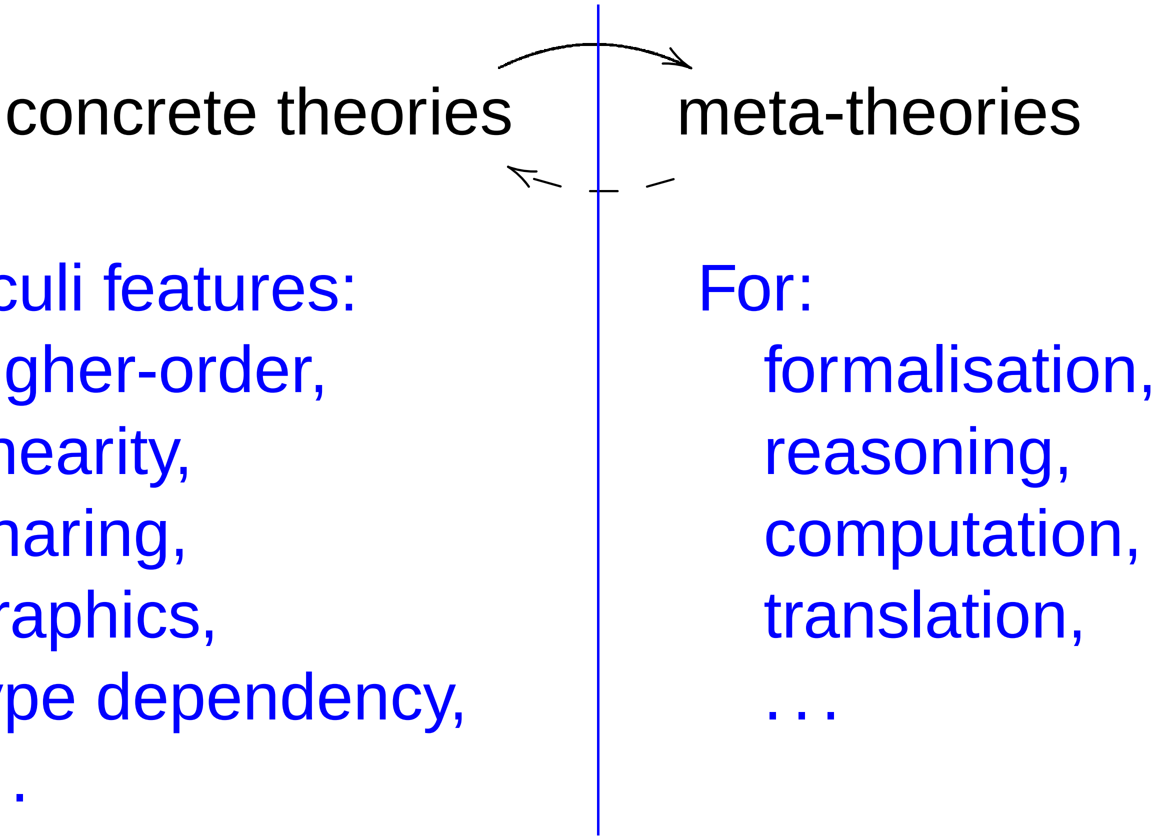
Calculi features:
- higher-order,
- linearity,
- sharing,
- graphics,
- type dependency,
- . . .

For:
- formalisation,
- reasoning,
- computation,
- translation,
- . . .

Programme

[1] mathematical
models

Context



concrete theories | meta-theories

Calculi features:
- higher-order,
- linearity,
- sharing,
- graphics,
- type dependency,
- . . .

For:
- formalisation,
- reasoning,
- computation,
- translation,
- . . .

Programme

[1] mathematical models $\overset{[2]}{\rightsquigarrow}$ meta-theories

This Talk
Part I

Mathematical *algebraic* framework and methodology for the synthesis of deduction systems for equational reasoning and computation by rewriting.

This Talk
Part I

Mathematical *algebraic* framework and methodology for the synthesis of deduction systems for equational reasoning and computation by rewriting.

- *Semantics*
  - Algebraic model theory
- *Syntax*
  - Initial-algebra semantics (⇒ compositionality)
  - structural recursion
  - induction principle
  - theory of translations

This Talk
Part II

TLCA

Typed Lambda Calculi:
- [simply-typed] λ-calculus,
- Martin Löf type theory,
- . . .

? What are Typed Lambda Calculi?

This Talk
Part II

TLCA

Typed Lambda Calculi:

[simply-typed] λ-calculus,
Martin Löf type theory,
. . .

$\boxed{?}$ What are Typed Lambda Calculi?

$\boxed{!}$ Simple Type Theories are
Second-Order Equational Presentations!

Part I

Equational Meta-Logic

Universal Algebra

Syntax	Semantics

Universal Algebra

Syntax	Semantics
signatures: $\Sigma = \{ \Sigma_n \in \mathcal{S}\!\mathit{et} \}_{n \in \mathbb{N}}$	

Universal Algebra

Syntax	Semantics
signatures: $\Sigma = \{ \Sigma_n \in \mathcal{S}\boldsymbol{et} \}_{n \in \mathbb{N}}$	
	algebras: $\underline{A} = \{ \Sigma_n \times A^n \to A \}_{n \in \mathbb{N}}$

Universal Algebra

Syntax	Semantics
signatures: $\Sigma = \{\, \Sigma_n \in \mathbf{Set} \,\}_{n \in \mathbb{N}}$	
	algebras: $\underline{A} = \{\, \Sigma_n \times A^n \to A \,\}_{n \in \mathbb{N}}$
	free constructions: $\begin{tikzcd} V \arrow[r] \arrow[dr, "\forall\, \rho"'] & T(V) \arrow[d, dashed, "\exists!\, \rho^{\#}"] \\ & \underline{A} \end{tikzcd}$

Universal Algebra

Syntax	Semantics
signatures: $\Sigma = \{ \Sigma_n \in \mathbf{Set} \}_{n \in \mathbb{N}}$	
	algebras: $\underline{A} = \{ \Sigma_n \times A^n \to A \}_{n \in \mathbb{N}}$
	free constructions: $\begin{tikzcd} V \arrow[r] \arrow[dr, "\forall \rho"'] & T(V) \arrow[d, dashed, "\exists! \, \rho^{\#}"] \\ & \underline{A} \end{tikzcd}$
terms, variables	

Universal Algebra

Syntax

signatures:

$$\Sigma = \{\, \Sigma_n \in \mathbf{\mathcal{S}et} \,\}_{n \in \mathbb{N}}$$

terms, variables, and substitution:

$$\begin{array}{ccc} T(V) \times \underline{A}^{V} & \to & \underline{A} \\ t, \rho & \mapsto & t[\rho] \end{array}$$

Semantics

algebras:

$$\underline{A} = \{\, \Sigma_n \times A^n \to A \,\}_{n \in \mathbb{N}}$$

free constructions:

$$\begin{tikzcd} V \arrow[r] \arrow[dr, "\forall\, \rho"'] & T(V) \arrow[d, dashed, "\exists!\, \rho^{\#}"] \\ & \underline{A} \end{tikzcd}$$

Universal Algebra

Syntax

signatures:

$$\Sigma = \{\, \Sigma_n \in \boldsymbol{\mathcal{S}et} \,\}_{n \in \mathbb{N}}$$

terms, variables, and substitution:

$$\begin{array}{ccc} T(V) \times \underline{A}^V & \to & \underline{A} \\ t, \rho & \mapsto & t[\rho] \end{array}$$

equations:

$$V \vdash t \equiv t'$$

Semantics

algebras:

$$\underline{A} = \{\, \Sigma_n \times A^n \to A \,\}_{n \in \mathbb{N}}$$

free constructions:

$$\begin{tikzcd} V \arrow[r] \arrow[dr, "\forall\, \rho"'] & T(V) \arrow[d, dashed, "\exists!\, \rho^{\#}"] \\ & \underline{A} \end{tikzcd}$$

Universal Algebra

Syntax

signatures:

$$\Sigma = \{\, \Sigma_n \in \mathcal{S}\!et \,\}_{n \in \mathbb{N}}$$

terms, variables, and substitution:

$$\begin{array}{ccc} T(V) \times \underline{A}^V & \to & \underline{A} \\ t, \rho & \mapsto & t[\rho] \end{array}$$

equations:

$$V \vdash t \equiv t'$$

Semantics

algebras:

$$\underline{A} = \{\, \Sigma_n \times A^n \to A \,\}_{n \in \mathbb{N}}$$

free constructions:

$$\begin{tikzcd} V \arrow[r] \arrow[dr, "\forall\, \rho"'] & T(V) \arrow[d, dashed, "\exists!\ \rho^{\#}"] \\ & \underline{A} \end{tikzcd}$$

validity:

$$\underline{A} \models t \equiv t'$$

iff $\forall \rho \in \underline{A}^V .\, t[\rho] = t'[\rho]$

Birkhoff's Deduction Problem

Devise a deduction system such that

$t \equiv t'$ is derivable from a set of equations $\mathcal{E}$

soundness $\Big\Downarrow$ $\Big\Uparrow$ completeness

for all $\underline{A} \models \mathcal{E}$, $\underline{A} \models t \equiv t'$

Birkhoff's Deduction Problem

Devise a deduction system such that

$t \equiv t'$ is derivable from a set of equations $\mathcal{E}$

soundness $\Downarrow$ $\Uparrow$ completeness

for all $\underline{A} \models \mathcal{E}$, $\underline{A} \models t \equiv t'$

Birkhoff's Equational Logic of Universal Algebra

$$\frac{(t \equiv t') \in \mathcal{E}}{t \equiv t'}$$

$$\frac{t_i \equiv t_i' \quad (i = 1, \ldots, n)}{f(t_1, \ldots, t_n) \equiv f(t_1', \ldots, t_n')} \quad (f : n)$$

$$\frac{t \equiv t'}{t[\rho] \equiv t'[\rho]} \quad (\rho \text{ a substitution})$$

Analysis of Universal Algebra

Syntax	Semantics
signatures: $\Sigma = \{ \Sigma_n \in \mathbf{Set} \}_{n \in \mathbb{N}}$	algebras: $\underline{A} = \{ \Sigma_n \times A^n \to A \}_{n \in \mathbb{N}}$ free constructions: $\begin{tikzcd} V \arrow[r] \arrow[dr, "\forall \rho"'] & T(V) \arrow[d, dashed, "\exists!\, \rho^{\#}"] \\ & \underline{A} \end{tikzcd}$
terms, variables, and substitution: $\begin{array}{ccc} T(V) \times \underline{A}^V & \to & \underline{A} \\ t, \rho & \mapsto & t[\rho] \end{array}$ equations: $V \vdash t \equiv t'$	validity: $\underline{A} \models t \equiv t'$ iff $\forall \rho \in \underline{A}^V .\, t[\rho] = t'[\rho]$

Monadic Algebra

Syntax	Semantics
	$\mathbb{T}$ a strong monad on a smc category algebras: $\underline{A} = (\mathsf{T}A \to A)$

Monadic Algebra

Generalised Syntax	Semantics
	$\mathbb{T}$ a strong monad on a smc category
	algebras: $\underline{A} = (TA \to A)$
generalised terms: $t : I \to TV$ (Kleisli maps)	

Monadic Algebra

Generalised Syntax	Semantics
	$\mathbb{T}$ a strong monad on a smc category
	algebras: $\underline{A} = (TA \to A)$
generalised terms: $t : I \to TV$ (Kleisli maps)	
variables: $V \to TV$	

Monadic Algebra

Generalised Syntax	Semantics
	$\mathbb{T}$ a strong monad on a smc category
	algebras: $\underline{A} = (TA \to A)$
generalised terms: $t : I \to TV$ (Kleisli maps)	
variables: $V \to TV$	
substitution: $\sigma : TV \otimes [V, \underline{A}] \to \underline{A}$	

Monadic Algebra

Generalised Syntax	Semantics
	$\mathbb{T}$ a strong monad on a smc category
	algebras: $\underline{A} = (TA \to A)$
generalised terms: $t : I \to TV$ (Kleisli maps)	
variables: $V \to TV$	
substitution: $\sigma : TV \otimes [V, \underline{A}] \to \underline{A}$	
	interpretation: $\begin{tikzcd} I \otimes [V, \underline{A}] \arrow[r, "[\![t]\!]"] \arrow[d, "t \otimes id"'] & \underline{A} \\ TV \otimes [V, \underline{A}] \arrow[ur, "\sigma"'] & \end{tikzcd}$

Monadic Algebra

Generalised Syntax	Semantics
	$\mathbb{T}$ a strong monad on a smc category
	algebras: $\underline{A} = (TA \to A)$
generalised terms: $t : I \to TV$ (Kleisli maps)	
variables: $V \to TV$	
substitution: $\sigma : TV \otimes [V, \underline{A}] \to \underline{A}$	
	interpretation: $\begin{tikzcd} I \otimes [V, \underline{A}] \arrow[r, "{[\![t]\!]}"] \arrow[d, "t \otimes id"'] & \underline{A} \\ TV \otimes [V, \underline{A}] \arrow[ur, "\sigma"'] & \end{tikzcd}$
generalised equations: $t \equiv t' : I \to TV$	

Monadic Algebra

<table>
<tr><th>Generalised Syntax</th><th>Semantics</th></tr>
<tr><td></td><td>$\mathbb{T}$ a strong monad
on a smc category</td></tr>
<tr><td></td><td>algebras:
$\underline{A} = (TA \to A)$</td></tr>
<tr><td>generalised terms:
$t : I \to TV$
(Kleisli maps)</td><td></td></tr>
<tr><td>variables:
$V \to TV$</td><td></td></tr>
<tr><td>substitution:
$\sigma : TV \otimes [V, \underline{A}] \to \underline{A}$</td><td></td></tr>
<tr><td></td><td>interpretation:

$$\begin{tikzcd}
I \otimes [V, \underline{A}] \arrow[r, "[\![t]\!]"] \arrow[d, "t \otimes id"'] & \underline{A} \\
TV \otimes [V, \underline{A}] \arrow[ru, "\sigma"'] &
\end{tikzcd}$$

</td></tr>
<tr><td>generalised equations:
$t \equiv t' : I \to TV$</td><td></td></tr>
<tr><td></td><td>validity:
$\underline{A} \models t \equiv t'$iff $[\![t]\!] = [\![t']\!]$</td></tr>
</table>

Deduction Problem

Devise a deduction system such that

$\mathrm{t} \equiv \mathrm{t}' : \mathrm{I} \to \mathsf{TV}$ is derivable from a set of generalised equations $\mathcal{E}$

soundness $\Downarrow$ $\Uparrow$ completeness

for all $\underline{A} \models \mathcal{E}$, $\underline{A} \models \mathrm{t} \equiv \mathrm{t}'$

Equational Meta-Logic Rules

Equational Meta-Logic Rules

$$(\text{Subst})\ \frac{\begin{array}{c} t_1 \equiv t_1' : U \to TV \\ t_2 \equiv t_2' : V \to TW \end{array}}{t_1[t_2] \equiv t_1'[t_2'] : U \to TW}$$

Equational Meta-Logic Rules

$$(\textsf{Subst})\ \frac{\begin{array}{c} t_1 \equiv t_1' : U \to TV \\ t_2 \equiv t_2' : V \to TW \end{array}}{t_1[t_2] \equiv t_1'[t_2'] : U \to TW}$$

$$(\textsf{LocChar})\ \frac{\begin{array}{c} \{\, e_i : U_i \to U \,\}_{i \in I} \text{ a cover} \\ t\, e_i \equiv t'\, e_i : U_i \to TV \quad (i \in I) \end{array}}{t \equiv t' : U \to TV}$$

Equational Meta-Logic Rules

$$(\text{Subst})\ \frac{\begin{array}{c} t_1 \equiv t_1' : U \to TV \\ t_2 \equiv t_2' : V \to TW \end{array}}{t_1[t_2] \equiv t_1'[t_2'] : U \to TW}$$

$$(\text{LocChar})\ \frac{\begin{array}{c} \{\, e_i : U_i \to U \,\}_{i \in I} \text{ a cover} \\ t\, e_i \equiv t'\, e_i : U_i \to TV \quad (i \in I) \end{array}}{t \equiv t' : U \to TV}$$

$$(\text{Ext})\ \frac{t \equiv t' : U \to TV}{\langle I \rangle t \equiv \langle I \rangle t' : I \otimes U \to T(I \otimes V)}$$

Soundness

If $t \equiv t' : I \to TV$ is derivable from $\mathcal{E}$

then $\underline{A} \models t \equiv t'$, for all $\underline{A} \models \mathcal{E}$.

Soundness

If $t \equiv t' : I \to TV$ is derivable from $\mathcal{E}$

then $\underline{A} \models t \equiv t'$, for all $\underline{A} \models \mathcal{E}$.

Internal Soundness and Completeness

For

$\widetilde{TV}$ the free algebra satisfying $\mathcal{E}$

and

$q : TV \to \widetilde{TV}$ the associated quotient map,

the following are equivalent:

1. $\underline{A} \models t \equiv t' : U \to TV$, for all $\underline{A} \models \mathcal{E}$
2. $\widetilde{TV} \models t \equiv t' : U \to TV$
3. $q\,t = q\,t' : U \to \widetilde{TV}$

Reconstruction of Birkhoff's Equational Logic

$$\mathcal{S}\boldsymbol{et}$$

syntactic structure

= signature: $\Sigma = \{\, \Sigma_n \in \mathcal{S}\boldsymbol{et} \,\}_{n \in \mathbb{N}}$

Reconstruction of Birkhoff's Equational Logic

$$\begin{tikzcd}
\mathbf{\Sigma\text{-}Alg} \arrow[d, shift left=3] \\
\mathbfit{Set} \arrow[u, shift left=3] \arrow[u, phantom, "\dashv"] \arrow[loop left, "\mathbb{T}_\Sigma"']
\end{tikzcd}$$

$$\mathsf{T}_\Sigma(X) \;\cong\; X + \coprod_{n\in\mathbb{N}} \Sigma_n \times (\mathsf{T}_\Sigma X)^n$$

interpretation

syntactic structure

= signature: $\Sigma = \{\, \Sigma_n \in \mathbfit{Set} \,\}_{n\in\mathbb{N}}$

Reconstruction of Birkhoff's Equational Logic

$$\begin{tikzcd}
\mathbf{\Sigma\text{-}Alg} \arrow[d, shift left=2ex] \arrow[d, phantom, "\dashv"] \\
\mathcal{S}et \arrow[u, shift left=2ex] \arrow[loop left, "\mathbb{T}_\Sigma"]
\end{tikzcd}$$

$$\mathsf{T}_\Sigma(\mathsf{X}) \;\cong\; \mathsf{X} + \coprod_{n \in \mathbb{N}} \Sigma_n \times (\mathsf{T}_\Sigma \mathsf{X})^n$$

interpretation

concretion

syntactic structure

= signature: $\Sigma = \{\, \Sigma_n \in \mathcal{S}et \,\}_{n \in \mathbb{N}}$

Equational Logic

$$\mathsf{V} \vdash \mathsf{t} \equiv \mathsf{t}'$$

Substitution Rule

$$\text{(Subst)}\ \dfrac{V \vdash s \equiv s' \qquad \text{(LocChar)}\ \dfrac{W \vdash t_x \equiv t'_x \quad (x \in V)}{W \vdash \{\, t_x \,\}_{x \in V} \equiv \{\, t'_x \,\}_{x \in V}}}{W \vdash s[t_x/x]_{x \in V} \equiv s'[t'_x/x]_{x \in V}}$$

Reconstruction of Goguen and Meseguer's Multi-Sorted Equational Logic

$\mathcal{S}et^{\mathsf{S}}$

syntactic structure

= signature: $\Sigma = \{\, \Sigma_\sigma \in \mathcal{S}et^{\mathsf{S}} \,\}_{\sigma \in \mathsf{S}^*}$

General method for the extension from the mono-sorted to the multi-sorted case.

Reconstruction of Goguen and Meseguer's Multi-Sorted Equational Logic

$$\begin{tikzcd}
\Sigma\text{-}\mathbf{Alg} \arrow[d, shift left] \arrow[d, phantom, "\dashv"] \\
\mathcal{S}et^{\mathsf{S}} \arrow[u, shift left] \arrow[loop left, "\mathbb{T}_\Sigma"]
\end{tikzcd}$$

$$(\mathsf{T}_\Sigma \mathsf{X})_{\mathsf{s}} \cong \mathsf{X}_{\mathsf{s}} + \coprod_{\sigma=(\mathsf{s}_1 \ldots \mathsf{s}_n) \in \mathsf{S}^*} \Sigma_{\sigma,\mathsf{s}} \times \prod_{i=1}^{n} (\mathsf{T}_\Sigma \mathsf{X})_{\mathsf{s}_i}$$

interpretation

syntactic structure

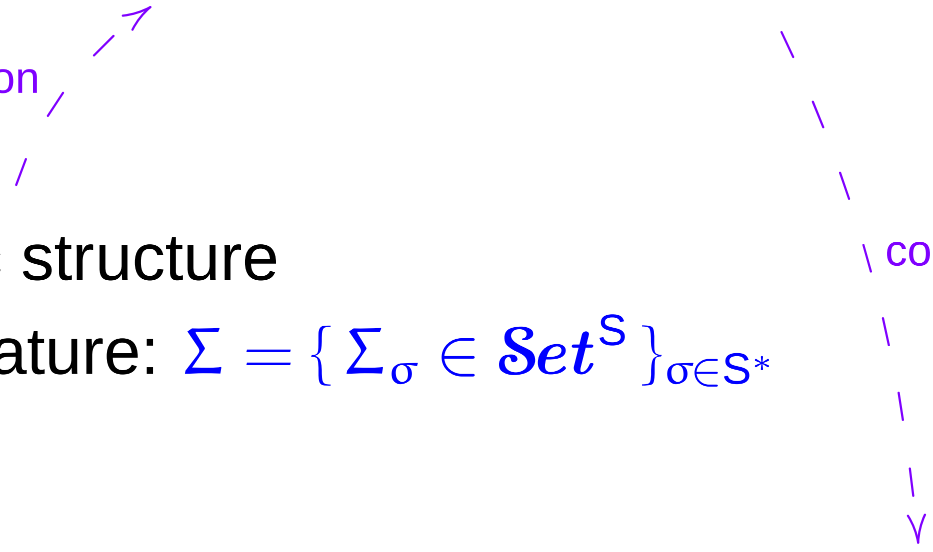
= signature: $\Sigma = \{ \Sigma_\sigma \in \mathcal{S}et^{\mathsf{S}} \}_{\sigma \in \mathsf{S}^*}$

General method for the extension from the mono-sorted to the multi-sorted case.

Reconstruction of Goguen and Meseguer's Multi-Sorted Equational Logic

$$\begin{tikzcd}
\mathbf{\Sigma\text{-}Alg} \arrow[d, shift left] \arrow[d, phantom, "\dashv"] \\
\mathcal{S}et^{S} \arrow[u, shift left] \arrow[loop left, "\mathbb{T}_\Sigma"]
\end{tikzcd}$$

$$(T_\Sigma X)_s \;\overset{\sim}{=}\; X_s + \coprod_{\sigma=(s_1\ldots s_n)\in S^*} \Sigma_{\sigma,s} \times \prod_{i=1}^{n} (T_\Sigma X)_{s_i}$$



interpretation

concretion

syntactic structure

= signature: $\Sigma = \{\, \Sigma_\sigma \in \mathcal{S}et^{S} \,\}_{\sigma\in S^*}$

Equational Logic

$$\vec{x} : \vec{\sigma} \vdash t \equiv t' : s$$

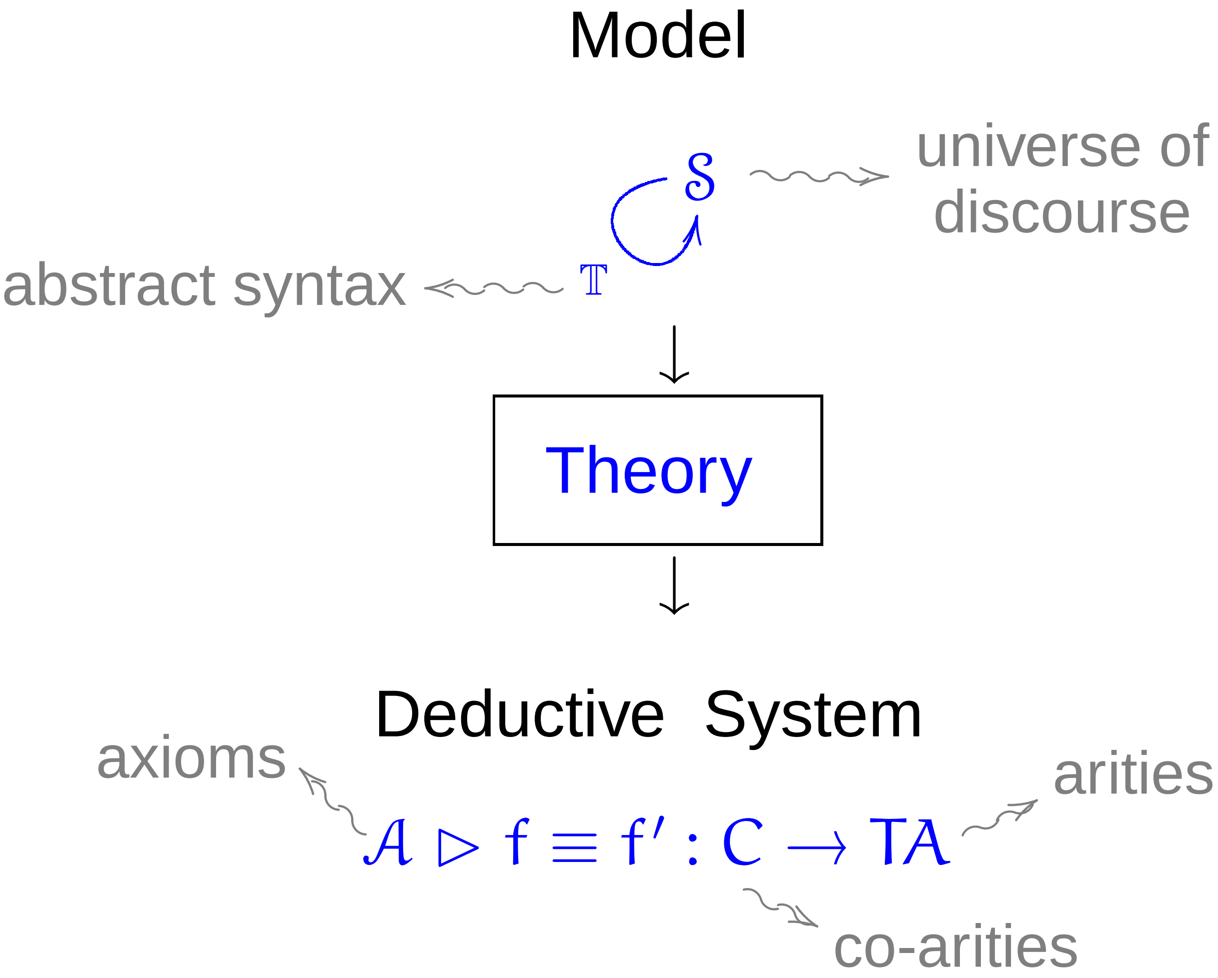
General method for the extension from the mono-sorted to the multi-sorted case.

Methodology

Model

$$\begin{tikzcd}
 & \mathcal{S} \arrow[loop left] \arrow[r, squiggly] & \text{universe of discourse} \\
\text{abstract syntax} & \mathbb{T} \arrow[l, squiggly] &
\end{tikzcd}$$

Methodology



Model

$\mathbb{T}$ $\mathcal{S}$

abstract syntax

universe of discourse

Theory

Deductive System

$$\mathcal{A} \rhd f \equiv f' : C \to TA$$

axioms

arities

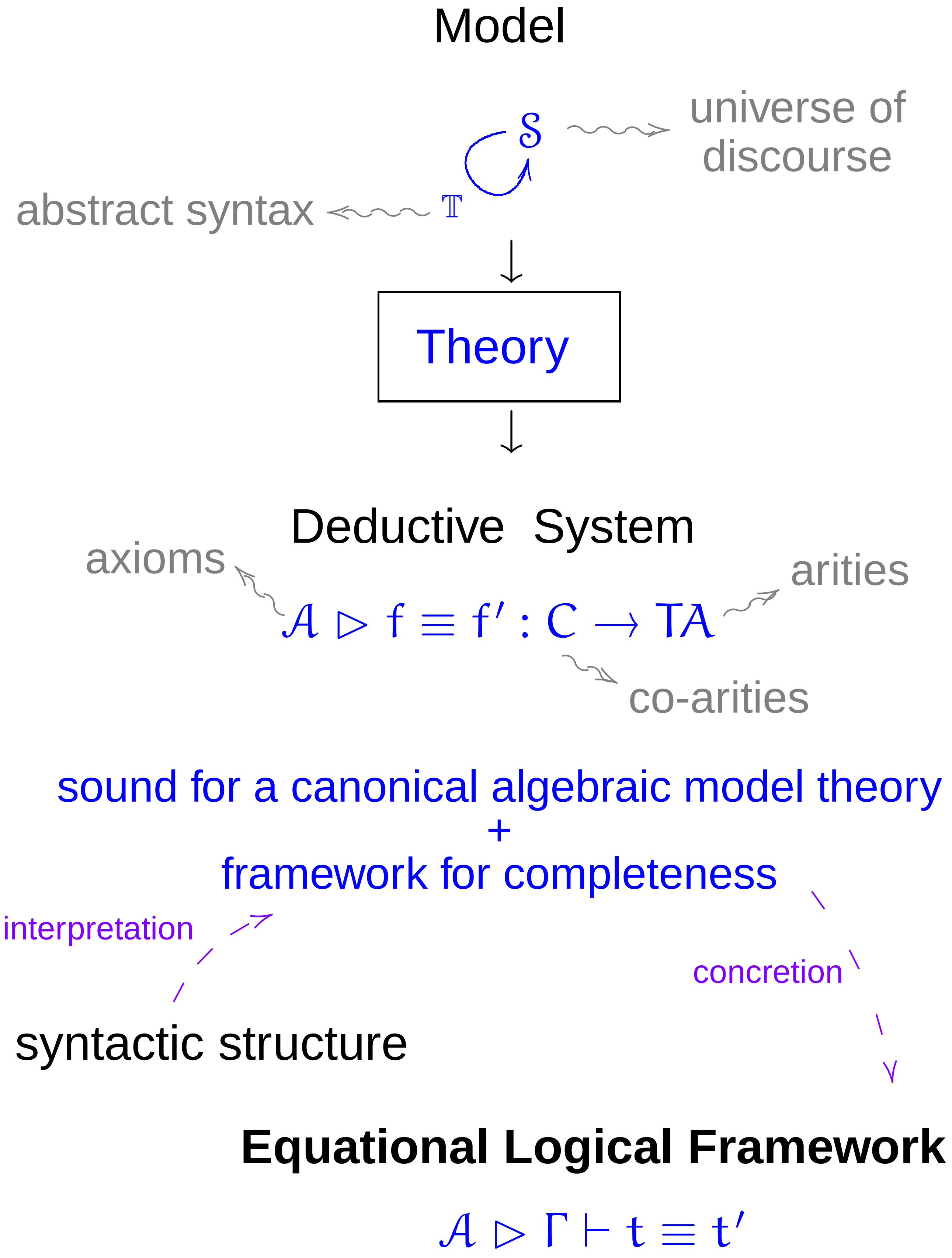
co-arities

sound for a canonical algebraic model theory

+

framework for completeness

Methodology



Model

$\mathcal{S}$ universe of discourse

abstract syntax $\mathbb{T}$

Theory

Deductive System

axioms arities

$\mathcal{A} \rhd f \equiv f' : C \to TA$

co-arities

sound for a canonical algebraic model theory
+
framework for completeness

interpretation

concretion

syntactic structure

Equational Logical Framework

$\mathcal{A} \rhd \Gamma \vdash t \equiv t'$

Part II

The Universal Algebra of Simple Type Theory

Simply Typed Theories

types	algebraic
terms	algebraic with binding

Simply Typed Theories

	algebraic	simply typed	dependently typed
		theories	
types	unstructured	algebraic	algebraic with binding
terms	algebraic	algebraic with binding	algebraic with binding

- The paradigmatic simple type theory:

  λ-calculus

  $$(\beta) \quad (\lambda x.\, M)\, N = M[N/x]$$

  $$(\eta) \quad \lambda x.\, M\, x = M \qquad \big(x \notin \mathrm{FV}(M)\big)$$

- Simply-typed λ-calculus, computational λ-calculus, . . .

- The paradigmatic simple type theory:

  λ-calculus

  $$(\beta) \quad (\lambda x.\, M)\, N = M[N/x]$$

  $$(\eta) \quad \lambda x.\, M\, x = M \qquad \big(x \notin \mathrm{FV}(M)\big)$$

- Simply-typed λ-calculus, computational λ-calculus, . . .

- The syntactic theory should account for:
  - variables and meta-variables
  - variable binding and α-equivalence
  - capture-avoiding and meta substitution
  - mono and multi sorting

Algebraic Model

syntactic structure =

- arities: an operator of arity $\vec{n} = (n_1 \dots n_k)$ takes k arguments, respectively binding n_i variables.

Algebraic Model

finite sets (contexts)
and functions (renamings)

$\mathcal{S}et^{\mathbb{F}}$

$X \in \mathcal{S}et^{\mathbb{F}}$ is a functor $\begin{cases} X\Gamma \ (\Gamma \in \mathbb{F}) \\ \mathbb{F}(\Gamma, \Delta) \to \mathcal{S}et(X\Gamma, X\Delta) \end{cases}$

E.g. the object of variables is $V\Gamma = \Gamma$

syntactic structure =

- arities: an operator of arity $\vec{n} = (n_1 \ldots n_k)$ takes k arguments, respectively binding n_i variables.

Algebraic Model

finite sets (contexts)
and functions (renamings)

$\mathcal{S}et^{\mathbb{F}}$

$X \in \mathcal{S}et^{\mathbb{F}}$ is a functor $\begin{cases} X\Gamma \ (\Gamma \in \mathbb{F}) \\ \mathbb{F}(\Gamma, \Delta) \to \mathcal{S}et(X\Gamma, X\Delta) \end{cases}$

E.g. the object of variables is $V\Gamma = \Gamma$

syntactic structure =

- arities: an operator of arity $\vec{n} = (n_1 \ldots n_k)$ takes k arguments, respectively binding n_i variables.
- signature: $\Sigma = \{ \Sigma_{\vec{n}} \in \mathcal{S}et^{\mathbb{F}} \}_{\vec{n} \in \mathbb{N}^*}$

Algebraic Model

finite sets (contexts)
and functions (renamings)

$$\begin{tikzcd} \mathcal{S}et^{\mathbb{F}} \arrow[loop left, "\Sigma"] \end{tikzcd}$$

$$\Sigma(X) = \coprod_{\vec{n}=(n_1 \ldots n_k)\in\mathbb{N}^*} \Sigma_{\vec{n}} \times \prod_{i=1}^{k} X^{V^{n_i}}$$

$X \in \mathcal{S}et^{\mathbb{F}}$ is a functor $\begin{cases} X\Gamma \ (\Gamma \in \mathbb{F}) \\ \mathbb{F}(\Gamma, \Delta) \to \mathcal{S}et(X\Gamma, X\Delta) \end{cases}$

E.g. the object of variables is $V\Gamma = \Gamma$

interpretation

syntactic structure =

- arities: an operator of arity $\vec{n} = (n_1 \ldots n_k)$ takes k arguments, respectively binding n_i variables.
- signature: $\Sigma = \{\Sigma_{\vec{n}} \in \mathcal{S}et^{\mathbb{F}}\}_{\vec{n}\in\mathbb{N}^*}$

Algebraic Model

finite sets (contexts)
and functions (renamings)

$$\begin{tikzcd} \mathcal{S}et^{\mathbb{F}} \arrow[loop left, "\Sigma"] \end{tikzcd}$$

$$\Sigma(X) = \coprod_{\vec{n}=(n_1 \ldots n_k)\in\mathbb{N}^*} \Sigma_{\vec{n}} \times \prod_{i=1}^{k} X^{V^{n_i}}$$

$X \in \mathcal{S}et^{\mathbb{F}}$ is a functor $\begin{cases} X\Gamma \quad (\Gamma \in \mathbb{F}) \\ \mathbb{F}(\Gamma,\Delta) \to \mathcal{S}et(X\Gamma, X\Delta) \end{cases}$

E.g. the object of variables is $V\Gamma = \Gamma$

interpretation

syntactic structure =

- arities: an operator of arity $\vec{n} = (n_1 \ldots n_k)$ takes k arguments, respectively binding n_i variables.
- signature: $\Sigma = \{\, \Sigma_{\vec{n}} \in \mathcal{S}et^{\mathbb{F}} \,\}_{\vec{n}\in\mathbb{N}^*}$

\+

- **substitution**

Algebras with Substitution

(Σ-monoids)

- algebra structure:

$$\Sigma X \xrightarrow{\xi} X$$

- substitution structure:

$$\text{monoid} \quad V \xrightarrow{e} X \xleftarrow{m} X \bullet X$$

$$\left(\equiv \quad \begin{array}{c} \Gamma \longrightarrow X\Gamma \longleftarrow X\Delta \times (X\Gamma)^{\Delta} \\ \text{subject to the laws of substitution} \end{array} \right)$$

subject to the compatibility condition:

$$\begin{tikzcd}
\Sigma(X) \bullet X \arrow[r] \arrow[d, "\xi \bullet X"'] & \Sigma(X \bullet X) \arrow[r, "\Sigma m"] & \Sigma X \arrow[d, "\xi"] \\
X \bullet X \arrow[rr, "m"'] & & X
\end{tikzcd}$$

λ Pre-Models with Substitution

- Signature.

$$\begin{aligned} \Sigma_\lambda \quad &= \quad \mathsf{0} \,@\, \mathsf{0} && \text{(application)} \\ &\mid \quad \lambda(1) && \text{(abstraction)} \end{aligned}$$

- Algebras.

$$\mathsf{app} : \mathrm{L}^2 \to \mathrm{L} \qquad \mathsf{abs} : \mathrm{L}^{\mathrm{V}} \to \mathrm{L}$$

$$\mathsf{var} : \mathrm{V} \longrightarrow \mathrm{L} \longleftarrow \mathrm{L} \bullet \mathrm{L} : \mathsf{subst}$$

$$\begin{tikzcd}
\mathrm{L}^2 \bullet \mathrm{L} \arrow[r, "\cong"] \arrow[d, "\mathsf{app} \bullet \mathrm{L}"'] & (\mathrm{L} \bullet \mathrm{L})^2 \arrow[r, "\mathsf{subst}^2"] & \mathrm{L}^2 \arrow[d, "\mathsf{app}"] \\
\mathrm{L} \bullet \mathrm{L} \arrow[rr, "\mathsf{subst}"'] & & \mathrm{L}
\end{tikzcd}$$

$$\begin{tikzcd}
\mathrm{L}^{\mathrm{V}} \bullet \mathrm{L} \arrow[r] \arrow[d, "\mathsf{abs} \bullet \mathrm{L}"'] & (\mathrm{L} \bullet \mathrm{L})^{\mathrm{V}} \arrow[r, "\mathsf{subst}^{\mathrm{V}}"] & \mathrm{L}^{\mathrm{V}} \arrow[d, "\mathsf{abs}"] \\
\mathrm{L} \bullet \mathrm{L} \arrow[rr, "\mathsf{subst}"'] & & \mathrm{L}
\end{tikzcd}$$

Monadic Model

$$\begin{tikzcd} \Sigma\text{-}\mathbf{Mon} \arrow[d, shift left=2] \arrow[d, phantom, "\dashv"] \\ \mathcal{S}et^{\mathbb{F}} \arrow[u, shift left=2] \arrow[loop left, "\mathcal{M}_\Sigma"] \end{tikzcd}$$

Thm:

1. $\mathcal{M}_\Sigma(X) \cong V + X \bullet \mathcal{M}_\Sigma(X) + \Sigma(\mathcal{M}_\Sigma X)$

Monadic Model

$$\begin{tikzcd} \Sigma\text{-}\mathbf{Mon} \arrow[d, shift left=2] \arrow[d, phantom, "\dashv"] \\ \mathcal{S}et^{\mathbb{F}} \arrow[u, shift left=2] \arrow[loop below, "\mathcal{M}_\Sigma"] \end{tikzcd}$$

Thm:

1. $$\mathcal{M}_\Sigma(X) \cong V + X \bullet \mathcal{M}_\Sigma(X) + \Sigma(\mathcal{M}_\Sigma X)$$

2. For Σ induced by a binding signature,

   $\mathcal{M}_\Sigma$ is a strong monad .

concretion

Syntactic theory for variables, meta-variables, variable binding, α-equivalence, capture-avoiding substitution, meta-substitution.

Second-Order Syntactic Theory (I)

- Syntax:

  For X an object of meta-variables,

$$
\begin{array}{lll}
t \in \mathcal{M}_\Sigma(X)_\Gamma & & \\
\quad ::= \lfloor x \rfloor & & (x \in \Gamma) \\
\quad\ \ \mid \ \ M[t_1, \ldots, t_\ell] & & \left(\begin{array}{l} M \in X(\ell) \\ t_i \in (M_\Sigma X)_\Gamma \end{array} \right) \\
\quad\ \ \mid \ \ f\big((\vec{x_1})t_1, \ldots, (\vec{x_k})t_k\big) & & \left(\begin{array}{l} f \in \Sigma_{(|\vec{x_1}|\ldots|\vec{x_k}|)} \\ t_i \in (M_\Sigma X)_{\Gamma,\vec{x_i}} \end{array} \right)
\end{array}
$$

Second-Order Syntactic Theory (I)

- Syntax:

  For X an object of meta-variables,

$$
\begin{array}{lll}
t \in \mathcal{M}_\Sigma(X)_\Gamma & & \\
\quad ::= \lfloor x \rfloor & & (x \in \Gamma) \\
\quad \mid \; \mathsf{M}[t_1, \ldots, t_\ell] & & \left(\begin{array}{l} \mathsf{M} \in X(\ell) \\ t_i \in (\mathcal{M}_\Sigma X)_\Gamma \end{array} \right) \\
\quad \mid \; \mathsf{f}\big((\vec{x_1})t_1, \ldots, (\vec{x_k})t_k\big) & & \left(\begin{array}{l} \mathsf{f} \in \Sigma_{(|\vec{x_1}| \ldots |\vec{x_k}|)} \\ t_i \in (\mathcal{M}_\Sigma X)_{\Gamma, \vec{x_i}} \end{array} \right)
\end{array}
$$

- Capture-avoiding substitution:

$$
\mathcal{M}_\Sigma(X) \bullet \mathcal{M}_\Sigma(X) \longrightarrow \mathcal{M}_\Sigma(X)
$$

$$
\left(\equiv \mathcal{M}_\Sigma(X)_\Delta \times \big(\mathcal{M}_\Sigma(X)_\Gamma\big)^\Delta \longrightarrow \mathcal{M}_\Sigma(X)_\Gamma \right)
$$

Second-Order Syntactic Theory (I)

- Syntax:

  For X an object of meta-variables,

$$
\begin{array}{lll}
t \in \mathcal{M}_\Sigma(X)_\Gamma & & \\
\quad ::= \lfloor x \rfloor & & (x \in \Gamma) \\
\quad \mid \; M[t_1, \ldots, t_\ell] & & \left(\begin{array}{l} M \in X(\ell) \\ t_i \in (M_\Sigma X)_\Gamma \end{array} \right) \\
\quad \mid \; f\big((\vec{x_1})t_1, \ldots, (\vec{x_k})t_k\big) & & \left(\begin{array}{l} f \in \Sigma_{(|\vec{x_1}|\ldots|\vec{x_k}|)} \\ t_i \in (M_\Sigma X)_{\Gamma,\vec{x_i}} \end{array} \right)
\end{array}
$$

- Capture-avoiding substitution:

$$\mathcal{M}_\Sigma(X) \bullet \mathcal{M}_\Sigma(X) \longrightarrow \mathcal{M}_\Sigma(X)$$

$$\left(\equiv \mathcal{M}_\Sigma(X)_\Delta \times \big(\mathcal{M}_\Sigma(X)_\Gamma\big)^\Delta \longrightarrow \mathcal{M}_\Sigma(X)_\Gamma \right)$$

- Meta-substitution:

$$\mathcal{M}_\Sigma(X) \times \big(\mathcal{M}_\Sigma(Y)\big)^X \longrightarrow \mathcal{M}_\Sigma(Y)$$

$$\left(\equiv \mathcal{M}_\Sigma(X)_\Gamma \times \prod_{\ell \in \mathbb{N}} X(\ell) \Rightarrow \left((\mathcal{M}_\Sigma Y)^{V^\ell} \right)_\Gamma \to \mathcal{M}_\Sigma(Y)_\Gamma \right)$$

Second-Order Syntactic Theory (II)

- *Canonical specification* and derived *correct definition* of
  - variable renaming,
  - capture-avoiding simultaneous substitution,
  - meta-variable renaming,
  - meta-substitution.
- Canonical *algebraic model theory*.

Second-Order Equational Presentations

$$x_1, \ldots, x_n \,;\, M_1[m_1], \ldots, M_k[m_k] \vdash t \equiv t'$$

Second-Order Algebraic Models

$$[\![t]\!] = [\![t']\!] : V^n \times A^{V^{m_1}} \times \ldots \times A^{V^{m_k}} \to A$$

Second-Order Equational Presentations

$$x_1, \ldots, x_n \,;\, M_1[m_1], \ldots, M_k[m_k] \vdash t \equiv t'$$

λ-calculus

$$(\beta) \quad \mathsf{M}[1], \mathsf{N}[0] \vdash \lambda\big((x)\mathsf{M}[\lfloor x \rfloor]\big) \,@\, \mathsf{N}[\,] \equiv \mathsf{M}\big[\mathsf{N}[\,]\big]$$

$$(\eta) \quad \mathsf{M}[\,] \vdash \lambda\big((x)\ \mathsf{M}[\,] \,@\, \lfloor x \rfloor\big) \equiv \mathsf{M}[\,]$$

Second-Order Algebraic Models

$$[\![t]\!] = [\![t']\!] : V^n \times A^{V^{m_1}} \times \ldots \times A^{V^{m_k}} \to A$$

Second-Order Equational Presentations

$$x_1, \ldots, x_n \,;\, M_1[m_1], \ldots, M_k[m_k] \vdash t \equiv t'$$

λ-calculus

$$(\beta) \quad M[1], N[0] \vdash \lambda\big((x)M[\lfloor x \rfloor]\big) \mathbin{@} N[\,] \equiv M\big[N[\,]\big]$$

$$(\eta) \quad M[\,] \vdash \lambda\big((x)\; M[\,] \mathbin{@} \lfloor x \rfloor\big) \equiv M[\,]$$

Second-Order Algebraic Models

$$[\![t]\!] = [\![t']\!] : V^n \times A^{V^{m_1}} \times \ldots \times A^{V^{m_k}} \to A$$

λ-models with substitution

$$\begin{tikzcd}
L^V \times L \arrow[r] \arrow[d, "\mathsf{abs}\times\mathrm{id}"'] \arrow[dr, phantom, "(\beta)"] & L \bullet L \arrow[d, "\mathsf{subst}"] \\
L \times L \arrow[r, "\mathsf{app}"'] & L
\end{tikzcd}$$

$$\begin{tikzcd}
 & & L \arrow[dll] \arrow[d, "\mathrm{id}"] \\
(V \times L)^V \arrow[d, "(\mathsf{var}\times L)^V"'] & (\eta) & L \\
(L \times L)^V \arrow[rr, "\mathsf{app}^V"'] & & L^V \arrow[u, "\mathsf{abs}"']
\end{tikzcd}$$

Second-Order Equational Logic (I)

$$\text{(Subst)}\ \frac{\begin{array}{c} t_1 \equiv t_1' : U \to TV \\ t_2 \equiv t_2' : V \to TW \end{array}}{t_1[t_2] \equiv t_1'[t_2'] : U \to TW}$$

Second-Order Equational Logic (I)

$$(\mathrm{Subst})\ \frac{\begin{array}{c} t_1 \equiv t_1' : U \to TV \\ t_2 \equiv t_2' : V \to TW \end{array}}{t_1[t_2] \equiv t_1'[t_2'] : U \to TW}$$

Substitution Rule

$$\frac{\begin{array}{l} \vec{x}\,;\, M_1[m_1], \ldots, M_k[m_k] \vdash t \equiv u \\ \begin{array}{l} y_1^{(i)}, \ldots, y_{m_i}^{(i)}\,;\, N_1[n_1], \ldots, N_\ell[n_\ell] \\ \qquad \vdash t_i \equiv u_i \end{array} \qquad (i = 1, \ldots, k) \end{array}}{\begin{array}{l} \vec{x}\,;\, N_1[n_1], \ldots, N_\ell[n_\ell] \\ \quad \vdash\ t\{\, M_i := (y_1^{(i)}, \ldots, y^{(i)m_i}\,)\, t_i \,\} \\ \qquad\quad \equiv\ u\{\, M_i := (y_1^{(i)}, \ldots, y^{(i)m_i}\,)\, u_i \,\} \end{array}}$$

Second-Order Equational Logic (II)

$$(\mathsf{LocChar})\ \dfrac{\begin{array}{c}\{\, e_i : U_i \to U \,\}_{i\in I} \text{ a cover} \\ t\, e_i \equiv t'\, e_i : U_i \to \mathsf{TV} \quad (i \in I)\end{array}}{t \equiv t' : U \to \mathsf{TV}}$$

Local Character Rule

$$\imath : \{\, x_1, \ldots, x_k \,\} \rightarrowtail \{\, y_1, \ldots, y_\ell \,\}$$

$$\dfrac{y_1, \ldots, y_\ell \,;\, \ldots, M_i[m_i], \ldots \vdash t[\imath] \equiv u[\imath]}{x_1, \ldots, x_k \,;\, \ldots, M_i[m_i], \ldots \vdash t \equiv u}$$

Second-Order Equational Logic (III)

$$(\mathsf{Ext})\ \frac{t \equiv t' : U \to TV}{\langle I \rangle t \equiv \langle I \rangle t' : I \otimes U \to T(I \otimes V)}$$

Extension Rule

$$\frac{\vec{x}\,;\ \ldots, M_i[m_i], \ldots\ \vdash\ t_1 \equiv t_2}{\vec{x}, y_1, \ldots, y_\ell\,;\ \ldots, M_i[m_i + \ell], \ldots \vdash t_1{}^{\#} \equiv t_2{}^{\#}}$$

$$t^{\#} = t\{\, M_i := (z_1^{(i)}, \ldots, z_{m_i}^{(i)})\, M_i[z_1^{(i)}, \ldots, z_{m_i}^{(i)}, y_1, \ldots, y_\ell]\,\}$$

Algebraic Simple Type Theory

Algebraic model theory

&

Second-order equational logic

for second-order equational presentations

+

Theory of translations