Towards a Formal Semantics for FHM: Part 2

Joey Capper and Henrik Nilsson

School of Computer Science, University of Nottingham

FPLab Away Day, Buxton, 8th of July 2011

What are we trying to achieve?

Evaluation of models in FHM.

1. Normalisation of functional level terms
2. Reducing signal level terms
3. Handling of simulation runtime events.

What are we trying to achieve?

Evaluation of models in FHM.

1. Normalisation of functional level terms
2. Reducing signal level terms
3. Handling of simulation runtime events.

What are we trying to achieve?

Evaluation of models in FHM.

1. Normalisation of functional level terms
2. Reducing signal level terms
3. Handling of simulation runtime events.

Partial Evaluation

The evaluation should be partial in some sense.

1. Permit only simple symbolic simplification at the signal level, e.g:

   β-reduction of signal level products.

   $$fst(x, y) + z = 0 \quad \rightsquigarrow \quad x + z = 0$$

2. Switch blocks may depend on an event payload. For example, state transfer between structural configurations.

Partial Evaluation

The evaluation should be partial in some sense.

1. Permit only simple symbolic simplification at the signal level, e.g:

β-reduction of signal level products.

$$fst(x, y) + z = 0 \quad \rightsquigarrow \quad x + z = 0$$

2. Switch blocks may depend on an event payload. For example, state transfer between structural configurations.

Partial Evaluation

The evaluation should be partial in some sense.

1. Permit only simple symbolic simplification at the signal level, e.g:

   β-reduction of signal level products.

   $$fst(x, y) + z = 0 \quad \rightsquigarrow \quad x + z = 0$$

2. Switch blocks may depend on an event payload. For example, state transfer between structural configurations.

Discrete Semantics

We are only interested in the discrete aspects of FHM.

1. Not concerned with continuous semantics
2. Want to keep a clean divide between the discrete and continuous aspects, allowing the continuous aspects to specified independently.

Discrete Semantics

We are only interested in the discrete aspects of FHM.

1. Not concerned with continuous semantics
2. Want to keep a clean divide between the discrete and continuous aspects, allowing the continuous aspects to specified independently.

How do we achieve this?

We use Normalisation by Evaluation (NbE), but why?

1. Reduction free view of normalisation
2. Symbolic method, enabling partial evaluation
3. We get use Agda as the meta-language!

How do we achieve this?

We use Normalisation by Evaluation (NbE), but why?

1. Reduction free view of normalisation
2. Symbolic method, enabling partial evaluation
3. We get use Agda as the meta-language!

How do we achieve this?

We use Normalisation by Evaluation (NbE), but why?

1. Reduction free view of normalisation
2. Symbolic method, enabling partial evaluation
3. We get use Agda as the meta-language!

What is NbE?

1. Closely related to type-directed partial evaluation
2. Proceeds by interpreting terms into an appropriate model
3. Objects of the model are then reified back into the normal forms that represent them.

What is NbE?

1. Closely related to type-directed partial evaluation
2. Proceeds by interpreting terms into an appropriate model
3. Objects of the model are then reified back into the normal forms that represent them.

What is NbE?

1. Closely related to type-directed partial evaluation
2. Proceeds by interpreting terms into an appropriate model
3. Objects of the model are then reified back into the normal forms that represent them.

How do we know our normaliser is correct?

The correctness of normalisation can be specified in terms of the equational theory ($\sim_{\beta\eta}$) of the language.

Soundness

$$t \sim_{\beta\eta} t' \implies \mathit{norm}\ t = \mathit{norm}\ t'$$

Completeness

$$t \sim_{\beta\eta} \mathit{norm}\ t$$

How do we know our normaliser is correct?

The correctness of normalisation can be specified in terms of the equational theory ($\sim_{\beta\eta}$) of the language.

Soundness

$$t \sim_{\beta\eta} t' \implies norm\ t = norm\ t'$$

Completeness

$$t \sim_{\beta\eta} norm\ t$$

The language

We consider an equation based language embedded into the simply typed λ-calculus.

Syntax

$$
\begin{aligned}
t ::=\ & x \\
\mid\ & t_1\ t_2 \\
\mid\ & \lambda x.t \\
\mid\ & \textbf{sigrel}\ z\ \textbf{where}\ q \\
\mid\ & \ldots etc
\end{aligned}
\qquad
\begin{aligned}
q ::=\ & t \diamond s \\
\mid\ & s_1 = s_2 \\
\mid\ & \textbf{init}\ s_1 = s_2 \\
\mid\ & \ldots etc
\end{aligned}
$$

The language

With a signal level language as follows:

Syntax

$$
\begin{aligned}
s ::= {} & z & \mid \ & \textbf{suc}\ s \\
\mid \ & t & \mid \ & \textbf{fst}\ s \\
\mid \ & s_1 + s_2 & \mid \ & \textbf{snd}\ s \\
\mid \ & s_1 * s_2 & \mid \ & \textbf{pair}\ s_1\ s_2 \\
\mid \ & \textbf{zero} & \mid \ & \ldots etc
\end{aligned}
$$

Types

With a simple language of types:

Syntax

$$\begin{aligned} \tau &::= \tau_1 \dot{\rightarrow} \tau_2 \\ &\mid\ \mathsf{SR}\ \sigma \\ &\mid\ \mathsf{Nat} \end{aligned} \qquad \begin{aligned} \sigma &::= \top \\ &\mid\ \sigma_1 \dot{\times} \sigma_2 \\ &\mid\ \mathsf{Nat} \end{aligned}$$

Equational Theory

We need to extend the equational theory ($\sim_{\beta\eta}$):

β-convertibility at $s : \sigma$.

$$(\textbf{sigrel}\ z\ \textbf{where}\ q) \diamond s \sim_{\beta\eta} \ q[s/z]$$

η-convertibility at $t : SR\ \sigma$.

$$t \sim_{\beta\eta} \textbf{sigrel}\ z\ \textbf{where}\ (t \diamond z)$$

+ new congruence rules, and $\beta\eta$ for signal level products.

Equational Theory

We need to extend the equational theory ($\sim_{\beta\eta}$):

β-convertibility at $s : \sigma$.

$$(\textbf{sigrel } z \textbf{ where } q) \diamond s \sim_{\beta\eta} \; q[s/z]$$

η-convertibility at $t : SR\ \sigma$.

$$t \sim_{\beta\eta} \textbf{sigrel } z \textbf{ where } (t \diamond z)$$

+ new congruence rules, and $\beta\eta$ for signal level products.

Equational Theory

We need to extend the equational theory ($\sim_{\beta\eta}$):

β-convertibility at $s : \sigma$.

$$(\textbf{sigrel}\ z\ \textbf{where}\ q) \diamond s \sim_{\beta\eta} \ q[s/z]$$

η-convertibility at $t : SR\ \sigma$.

$$t \sim_{\beta\eta} \textbf{sigrel}\ z\ \textbf{where}\ (t \diamond z)$$

$+$ new congruence rules, and $\beta\eta$ for signal level products.

Term Representation

Some signatures:

Term representation.

$$\begin{aligned}
&\mathit{STerm}\ (\Gamma\ \Delta : \mathit{Ctx}) : (\sigma : \mathit{SType}) \rightarrow \mathit{Set} \\
&\mathit{Term}\ (\Gamma : \mathit{Ctx}) : (\tau : \mathit{Type}) \rightarrow \mathit{Set} \\
&\mathit{EqTerm}\ (\Gamma\ \Delta : \mathit{Ctx}) : \mathit{Set}
\end{aligned}$$

The Model

A type directed interpretation into the model (roughly!). Nrm is the redex-free representation of terms. First, the signal types.

Model interpretation

$$
\begin{aligned}
&\mathit{SVal} : (\Gamma\ \Delta : \mathit{Ctx}) \rightarrow \mathit{SType} \rightarrow \mathit{Set} \\
&\mathit{SVal}\ \Gamma\ \Delta\ \mathit{Unit} && = \mathit{SNrm}\ \Gamma\ \Delta\ \mathit{Unit} \\
&\mathit{SVal}\ \Gamma\ \Delta\ \mathrm{Nat} && = \mathit{SNrm}\ \Gamma\ \Delta\ \mathrm{Nat} \\
&\mathit{SVal}\ \Gamma\ \Delta\ (\sigma_1 \dot{\times} \sigma_2) && = \mathit{SVal}\ \Gamma\ \Delta\ \sigma_1 \times \mathit{SVal}\ \Gamma\ \Delta\ \sigma_2
\end{aligned}
$$

The Model

And the functional types.

Model interpretation

$$
\begin{array}{ll}
\mathit{Val} : (\Gamma : \mathit{Ctx}) \rightarrow \mathit{Type} \rightarrow \mathit{Set} & \\
\mathit{Val}\ \Gamma\ \mathrm{Nat} & = \mathit{Nrm}\ \Gamma\ \Delta\ \mathrm{Nat} \\
\mathit{Val}\ \Gamma\ (\tau_1 \dot{\rightarrow} \tau_2) & = \mathit{Val}\ \Gamma\ \tau_1 \rightarrow \mathit{Val}\ \Gamma\ \tau_2 \\
\mathit{Val}\ \Gamma\ (\mathit{SR}\ \sigma) & = \mathit{SVal}\ \Gamma\ \sigma \rightarrow \mathit{EqNrm}\ \Gamma\ \Delta
\end{array}
$$

It is now possible to give our interpreter.

Interpreter type signatures

$$
\begin{aligned}
[\![\cdot]\!]_s &: \mathit{STerm}\ \Gamma\ \Delta\ \sigma \rightarrow \mathit{Env}\ \Gamma \rightarrow \mathit{Env}\ \Delta \rightarrow \mathit{SVal}\ \Delta\ \sigma \\
[\![\cdot]\!] &: \mathit{Term}\ \Gamma\ \tau \rightarrow \mathit{Env}\ \Gamma \rightarrow \mathit{Val}\ \Gamma\ \tau
\end{aligned}
$$

Env is just an environment of values for each variable in the indexing context.

The final step is to take our representative in the model, and convert it back into a normal form.

Reification

$$\begin{aligned} \mathit{reify} &: \mathit{Val}\ \Gamma\ \tau \quad \rightarrow \mathit{Nrm}\ \Gamma\ \tau \\ \mathit{reify}_s &: \mathit{SVal}\ \Delta\ \sigma \rightarrow \mathit{SNrm}\ \Delta\ \sigma \end{aligned}$$

Composition of interpretation and reification is normalisation!

The handling of events is quite rudimentary at the moment. But maybe there is a better way? Coinduction maybe?

The end

Questions?

For further details, see our draft at `cs.nott.ac.uk/~jjc`