

Full Abstraction for Reduced ML

Nikos Tzevelekos
Oxford University Computing Laboratory

Joint work with Andrzej Murawski

FoSSaCS, York, March 2009

What this talk is about

Denotational semantics of Reduced ML (Reduced ML).

- Reduced ML (Stark '94) is a functional language with *nominal* integer references.
- Its denotational modelling had not been addressed before in a satisfactory way.
- We provide such a model in *Nominal Games*.

What this talk is about

Nominal references
Reduced ML: syntax
Reduced ML: typing rules
Reduced ML: reduction rules
Examples
Contextual equivalence
Examples
Full abstraction
Nominal Games
Strategies
Local state
Considerations
Name innocence
Restrictions on plays
Restrictions on strategies
Name blindness
Results
Conclusion

Names in programming languages are *atomic* constructs which can be:

- created fresh locally,
- compared for equality,
- passed around via function application.

...*Locality, Distinguishability, Mobility.*

These are the basic specifications, describing a very 'simple' nominal language, the ν -calculus (Pitts & Stark 93).

Other uses for names: π -calculus, Java, ML, etc.

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Names in programming languages are *atomic* constructs which can be:

- created fresh locally,
- compared for equality,
- passed around via function application.

...*Locality, Distinguishability, Mobility.*

These are the basic specifications, describing a very ‘simple’ nominal language, the ν -calculus (Pitts & Stark 93).

Other uses for names: π -calculus, Java, ML, etc.

Nominal references can also be:

- dereferenced and updated.

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Simply-typed λ -calculus with integers and references.

Assume a countably infinite set of names, \mathbb{A} , with elements a, b, \dots .

■ Types:

$$\theta ::= \text{unit} \mid \text{int} \mid \text{int ref} \mid \theta \rightarrow \theta$$

■ Terms:

$$\begin{aligned} M ::= & x \mid \lambda x.M \mid M M \mid () \mid \Omega \\ & \mid n \mid M \odot M \mid \text{if } M \text{ then } M \text{ else } M \\ & \mid a \mid \text{ref } M \mid !M \mid M := M \end{aligned}$$

$$V ::= n \mid () \mid a \mid x \mid \lambda x.M$$

- Typing judgments have the shape

$$u, \Gamma \vdash M : \theta$$

where u is a finite subset of \mathbb{A} .

$$\frac{}{u, \Gamma \vdash () : \text{unit}} \quad \frac{}{u, \Gamma \vdash \Omega : \text{unit}} \quad \frac{i \in \mathbb{Z}}{u, \Gamma \vdash i : \text{int}}$$

$$\frac{(x : \theta) \in \Gamma}{u, \Gamma \vdash x : \theta} \quad \frac{u, \Gamma \vdash M : \theta \rightarrow \theta' \quad u, \Gamma \vdash N : \theta}{u, \Gamma \vdash MN : \theta'} \quad \frac{u, \Gamma \oplus \{x : \theta\} \vdash M : \theta'}{u, \Gamma \vdash \lambda x^\theta. M : \theta \rightarrow \theta'}$$

$$\frac{u, \Gamma \vdash M_1 : \text{int} \quad u, \Gamma \vdash M_2 : \text{int}}{u, \Gamma \vdash M_1 \odot M_2 : \text{int}} \quad \frac{u, \Gamma \vdash M : \text{int} \quad u, \Gamma \vdash N_0, N_1 : \theta}{u, \Gamma \vdash \text{if } M \text{ then } N_1 \text{ else } N_0 : \theta}$$

- Typing judgments have the shape

$$u, \Gamma \vdash M : \theta$$

where u is a finite subset of \mathbb{A} .

$$\frac{a \in u}{u, \Gamma \vdash a : \text{intref}} \quad \frac{u, \Gamma \vdash M : \text{int}}{u, \Gamma \vdash \text{ref } M : \text{intref}}$$

$$\frac{u, \Gamma \vdash M : \text{intref}}{u, \Gamma \vdash !M : \text{int}}$$

$$\frac{u, \Gamma \vdash M : \text{intref} \quad u, \Gamma \vdash N : \text{int}}{u, \Gamma \vdash M := N : \text{unit}}$$

Reduction rules have the shape

$$s, M \longrightarrow s', M'$$

where $s, s' : \mathbb{A} \rightarrow_{\text{fin}} \mathbb{Z}$.

- $s, (\lambda x.M)V \longrightarrow M[V/x]$
- $s, \text{if } 0 \text{ then } N_1 \text{ else } N_0 \longrightarrow s, N_0$
- $s, \text{if } 1 \text{ then } N_1 \text{ else } N_0 \longrightarrow s, N_1$
- ...
- $s, \text{ref } i \longrightarrow s(a \mapsto i), a$ with $a \notin \text{dom}(s)$
- $s, a := i \longrightarrow s(a \mapsto i), ()$
- $s, !a \longrightarrow s, i$ with $s(a) = i$.

Notation:

- $M; N \equiv (\lambda x.N)M$, some x not free in N .
- $\text{new}(x, M) \text{ in } N \equiv (\lambda x^{\text{intref}}.N)(\text{ref } M)$

$$s, \text{new}(x, M) \text{ in } N \longrightarrow s', \text{new}(x, i) \text{ in } N \longrightarrow s'(a \mapsto i), N[a/x]$$

Notation:

- $M; N \equiv (\lambda x.N)M$, some x not free in N .
- $\text{new}(x, M) \text{ in } N \equiv (\lambda x^{\text{intref}}.N)(\text{ref } M)$

Examples

- Scope extrusion. E.g: $\lambda x. \text{ref } x : \text{int} \rightarrow \text{int ref}$.

Notation:

- $M; N \equiv (\lambda x.N)M$, some x not free in N .
- $\text{new}(x, M) \text{ in } N \equiv (\lambda x^{\text{intref}}.N)(\text{ref } M)$

Examples

- Scope extrusion. E.g: $\lambda x. \text{ref } x : \text{int} \rightarrow \text{int ref}$.
- Functions with local state. E.g. define $\text{Counter} : \text{unit} \rightarrow \text{int}$ as:

$$\text{new}(z, 0) \text{ in } (\lambda x^{\text{unit}}. z := !z + 1; !z)$$

Notation:

- $M; N \equiv (\lambda x.N)M$, some x not free in N .
- $\text{new}(x, M) \text{ in } N \equiv (\lambda x^{\text{intref}}.N)(\text{ref } M)$

Examples

- Scope extrusion. E.g: $\lambda x. \text{ref } x : \text{int} \rightarrow \text{int ref}$.
- Functions with local state. E.g. define $\text{Counter} : \text{unit} \rightarrow \text{int}$ as:

$$\text{new}(z, 0) \text{ in } (\lambda x^{\text{unit}}. z := !z + 1; !z)$$

- Name equality. Define $\text{Test} : \text{intref} \rightarrow \text{intref} \rightarrow \text{int}$ as:

$$\lambda x^{\text{intref}}. \lambda y^{\text{intref}}. \text{new}(z, 0), (x', !x), (y', !y) \text{ in}$$

$$x := 0; y := 1; (\text{if } !x \text{ then } z := 1 \text{ else } ());$$

$$x := !x'; y := !y'; !z$$

Contextual equivalence

Given $\vdash M : \text{unit}$ we write $M \Downarrow$ if, for some s ,

$$\emptyset, M \longrightarrow s, ()$$

Definition

Terms-in-context $\emptyset, \Gamma \vdash M_1 : \theta$ and $\emptyset, \Gamma \vdash M_2 : \theta$ are **equivalent** (written $\Gamma \vdash M_1 \cong M_2$) if

$$C[M_1] \Downarrow \iff C[M_2] \Downarrow$$

for any context $C[-]$ such that $\vdash C[M_1], C[M_2] : \text{unit}$.

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

$\vdash \text{new } (x, 0), (y, 0) \text{ in } M \cong \text{new } (y, 0), (x, 0) \text{ in } M$

$\vdash \text{new } (y, 0) \text{ in } (\lambda x^{\text{int ref}}. \text{Test}(x, y)) \cong \lambda x^{\text{int ref}}. 0$

$f : \text{int ref} \rightarrow \text{unit} \vdash \text{new } (x, 0) \text{ in } (fx ; x := 0 ; fx)$
 $\cong \text{new } (x, 0), (y, 0) \text{ in } (fx ; fy)$

What this talk is about

Nominal references

Reduced ML: syntax
Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

A denotational model for Reduced ML is **fully abstract** if

$$\Gamma \vdash M_1 \cong M_2 : \theta \iff \llbracket \Gamma \vdash M_1 : \theta \rrbracket = \llbracket \Gamma \vdash M_2 : \theta \rrbracket$$

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

A denotational model for Reduced ML is **fully abstract** if

$$\Gamma \vdash M_1 \cong M_2 : \theta \iff \llbracket \Gamma \vdash M_1 : \theta \rrbracket = \llbracket \Gamma \vdash M_2 : \theta \rrbracket$$

This had not been solved satisfactorily.

- Traditional denotational models:
*global/fixed scope (no locality/mobility),
soundness only.*
- Game semantics (Abramsky & McCusker '97):
“bad” variables.
- Nominal game semantics (Laird '04, Laird '08):
references to names, not integers.
- Nominal game semantics (AGMOS '04, Tz. '07):
FA via semantical quotienting.

What this talk is about

Nominal references

Reduced ML: syntax
Reduced ML: typing rules

Reduced ML:
reduction rules

Examples

Contextual
equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on
strategies

Name blindness

Results

Conclusion

- Two participants: O (environment) and P (program).
- Games specified by plays, i.e. *justified* sequences of *moves*. Moves can contain names, i.e. elements of \mathbb{A} .
- Moves selected from *arenas*. Examples:

$$\llbracket \text{unit} \rrbracket = \{ *P \} \quad \llbracket \text{int} \rrbracket = \{ i_P \mid i \in \mathbb{Z} \}$$

$$\llbracket \text{int ref} \rrbracket = \{ a_P \mid a \in \mathbb{A} \}$$

$$\llbracket \text{int ref} \rightarrow \text{unit} \rrbracket = \begin{array}{c} *P \\ \swarrow \\ a_O \\ \searrow \\ *P \end{array}$$

- Founded on (*linear*) *nominal sets* (Gabbay & Pitts '99).

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

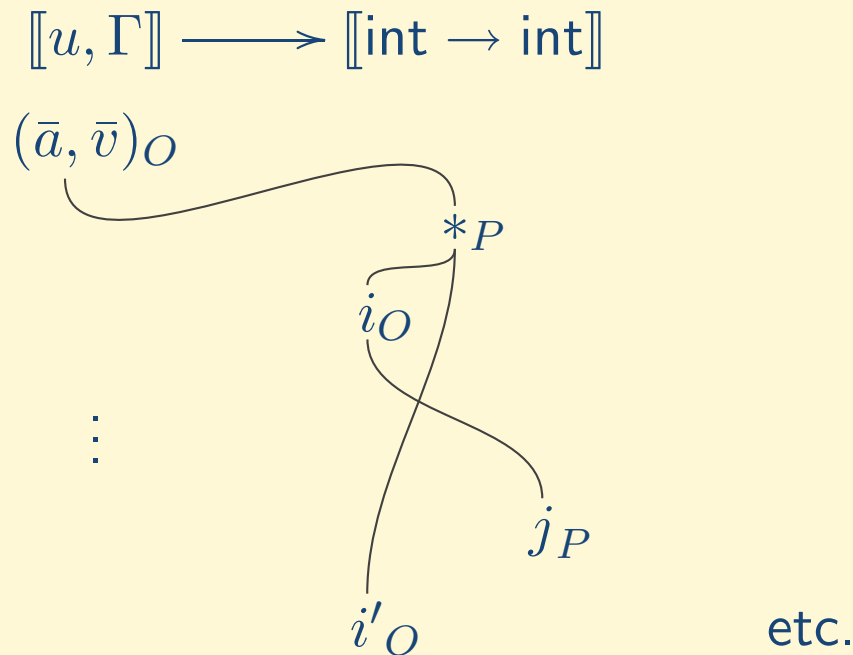
Name blindness

Results

Conclusion

- Programs are interpreted as *strategies* for P .
 - ◆ These are *deterministic* up to choice of fresh names,
 - ◆ saturated wrt name-permutations.

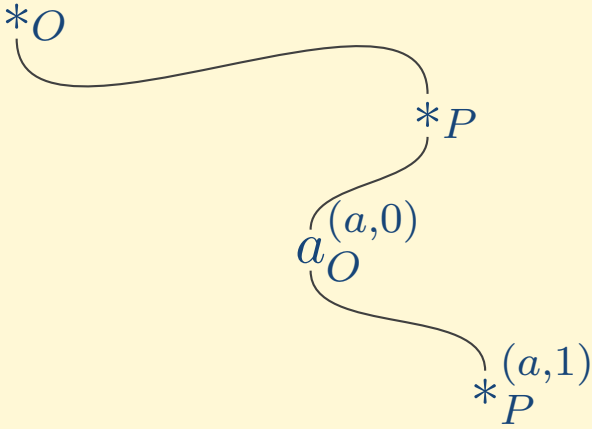
Example: how does $\llbracket u, \Gamma \vdash \lambda x.M : \text{int} \rightarrow \text{int} \rrbracket$ look like?



- What this talk is about
- Nominal references
- Reduced ML: syntax
- Reduced ML: typing rules
- Reduced ML: reduction rules
- Examples
- Contextual equivalence
- Examples
- Full abstraction
- Nominal Games
- Strategies**
- Local state
- Considerations
- Name innocence
- Restrictions on plays
- Restrictions on strategies
- Name blindness
- Results
- Conclusion

- We use moves attached with store:

$$[[\emptyset, \emptyset]] \longrightarrow [[\text{int ref} \rightarrow \text{unit}]]$$



Notation: $\widehat{*}_O \widehat{*}_P \widehat{a}_O^{(a,0)} \widehat{*}_P^{(a,1)}$

- Stores should not introduce fresh names (*frugality*).

- What this talk is about
- Nominal references
- Reduced ML: syntax
- Reduced ML: typing rules
- Reduced ML: reduction rules
- Examples
- Contextual equivalence
- Examples
- Full abstraction
- Nominal Games
- Strategies
- Local state**
- Considerations
- Name innocence
- Restrictions on plays
- Restrictions on strategies
- Name blindness
- Results
- Conclusion

Considerations:

- Which names are available to P ?
- Which names should appear in the store?

Examples

- Consider a function of type $\text{int} \rightarrow \text{int}$.

$$u, \Gamma \vdash \lambda x^{\text{int}}. M$$

Can it 'remember' x 's from different function calls?

- Now consider a function of type $\text{intref} \rightarrow \text{int}$.

$$u, \Gamma \vdash \lambda x^{\text{intref}}. N$$

Can it 'remember' x 's from different function calls?

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Name innocence

- Therefore, P cannot have full information of the history of a play.
- In particular, he can only see his own names and those of the current *closure*.
Put differently, P is *purely functional* namewise.
- But he can see everything else..

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

- Therefore, P cannot have full information of the history of a play.
- In particular, he can only see his own names and those of the current *closure*.
Put differently, P is *purely functional* namewise.
- But he can see everything else..

In game semantics:

current closure $\mapsto P$ -view

pure behaviour \mapsto *Innocence*

General innocence:

- Introduced by (Hyland & Ong '94).
- Captures PCF.

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

The P -view of a play s (written $\lceil s \rceil$) is the subplay corresponding to the computation that has been performed under (and up to) the current closure.

We define the set of a play's P -**available** names:

$$Av_P(s) = P(s) \cup \nu(\lceil s \rceil)$$

P -availability If P plays an O -name then it must have been P -available.

$$s'p^S \text{ a play and } a \in (\nu(p) \cap O(s')) \implies a \in Av_P(s')$$

P -storage Only P -available names appear in the store.

$$(s'm^S \text{ a play}) \implies \text{dom}(S) = Av_P(s'm^S)$$

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Restrictions on strategies

Conditions on plays:

- P cannot play unavailable names.
- P cannot change/use the values of unavailable names.

But he can still see them..

E.g. can a strategy σ_{bad} contain both the following plays?

$$\begin{array}{cccccc} * & * & a(a,0) & *(a,1) & a(a,0) & *(a,1) \\ O & P & O & P & O & P \end{array}$$

$$\begin{array}{cccccc} * & * & a(a,0) & *(a,1) & b(b,0) & *(b,0) \\ O & P & O & P & O & P \end{array}$$

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

- We can define a notion of **renamings** invisible to P . (Note these are not just name-permutations..)
- This induces an equivalence relation \sim^r on plays.

Definition

A strategy σ is **blind** if $s \in \sigma$ implies $s' \in \sigma$ for all $s' \sim^r s$.

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

- We can define a notion of **renamings** invisible to P .
(Note these are not just name-permutations..)
- This induces an equivalence relation \sim^r on plays.

Definition

A strategy σ is **blind** if $s \in \sigma$ implies $s' \in \sigma$ for all $s' \sim^r s$.

What about σ_{bad} ?

$$* \widehat{*} \overbrace{a^{(a,0)} *^{(a,1)} a^{(a,0)} *^{(a,1)}}^{\sim^r}$$

$$* \widehat{*} \overbrace{a^{(a,0)} *^{(a,1)} b^{(b,0)} *^{(b,0)}}^{\sim^r}$$

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

- We can define a notion of **renamings** invisible to P .
(Note these are not just name-permutations..)
- This induces an equivalence relation $\overset{r}{\sim}$ on plays.

Definition

A strategy σ is **blind** if $s \in \sigma$ implies $s' \in \sigma$ for all $s' \overset{r}{\sim} s$.

What about σ_{bad} ?

$$\begin{array}{c}
 \overset{\wedge}{*} \overset{\wedge}{*} \overset{\frown}{a(a,0)} \overset{\frown}{*(a,1)} \overset{\frown}{a(a,0)} \overset{\frown}{*(a,1)} \quad \overset{r}{\sim} \quad \overset{\wedge}{*} \overset{\wedge}{*} \overset{\frown}{a(a,0)} \overset{\frown}{*(a,1)} \overset{\frown}{b(b,0)} \overset{\frown}{*(b,1)} \\
 \overset{\wedge}{*} \overset{\wedge}{*} \overset{\frown}{a(a,0)} \overset{\frown}{*(a,1)} \overset{\frown}{b(b,0)} \overset{\frown}{*(b,0)}
 \end{array}$$

- What this talk is about
- Nominal references
- Reduced ML: syntax
- Reduced ML: typing rules
- Reduced ML: reduction rules
- Examples
- Contextual equivalence
- Examples
- Full abstraction
- Nominal Games
- Strategies
- Local state
- Considerations
- Name innocence
- Restrictions on plays
- Restrictions on strategies
- Name blindness**
- Results
- Conclusion

- We can define a notion of **renamings** invisible to P .
(Note these are not just name-permutations..)
- This induces an equivalence relation $\overset{r}{\sim}$ on plays.

Definition

A strategy σ is **blind** if $s \in \sigma$ implies $s' \in \sigma$ for all $s' \overset{r}{\sim} s$.

What about σ_{bad} ?

$$\begin{array}{c}
 \widehat{*} \widehat{*} \overset{\curvearrowright}{a(a,0)} \widehat{*} \overset{\curvearrowright}{(a,1)} \widehat{*} \overset{\curvearrowright}{b(b,0)} \widehat{*} \overset{\curvearrowright}{(b,1)} \\
 \widehat{*} \widehat{*} \overset{\curvearrowright}{a(a,0)} \widehat{*} \overset{\curvearrowright}{(a,1)} \widehat{*} \overset{\curvearrowright}{b(b,0)} \widehat{*} \overset{\curvearrowright}{(b,0)}
 \end{array}$$

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

- We can define a notion of **renamings** invisible to P . (Note these are not just name-permutations..)
- This induces an equivalence relation $\overset{r}{\sim}$ on plays.

Definition

A strategy σ is **blind** if $s \in \sigma$ implies $s' \in \sigma$ for all $s' \overset{r}{\sim} s$.

What about σ_{bad} ?

$$* \overset{\wedge}{*} \overset{\frown}{a(a,0)} \overset{\frown}{*(a,1)} \overset{\frown}{b(b,0)} \overset{\frown}{*(b,1)}$$

breaks determinacy!

$$* \overset{\wedge}{*} \overset{\frown}{a(a,0)} \overset{\frown}{*(a,1)} \overset{\frown}{b(b,0)} \overset{\frown}{*(b,0)}$$

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Proposition

There is a category \mathcal{G} having arenas as objects and blind strategies as arrows.

Proposition

\mathcal{G} is a sound model of Reduced ML.

Theorem

Finitary blind strategies are definable.

Theorem

Two Reduced ML terms are equivalent iff they generate the same sets of *complete symmetric* plays.

What this talk is about

Nominal references

Reduced ML: syntax
Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Notes:

- The fully abstract model of Reduced ML is effectively presentable.
- Nevertheless, Reduced ML is undecidable (Murawski '03).

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Notes:

- The fully abstract model of Reduced ML is effectively presentable.
- Nevertheless, Reduced ML is undecidable (Murawski '03).

What has been achieved?

- We have a handle on program equivalence in Reduced ML.
- We have new notions and intuitions in nominal games (name-availability, blindness).

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion

Notes:

- The fully abstract model of Reduced ML is effectively presentable.
- Nevertheless, Reduced ML is undecidable (Murawski '03).

What has been achieved?

- We have a handle on program equivalence in Reduced ML.
- We have new notions and intuitions in nominal games (name-availability, blindness).

THANKS!

What this talk is about

Nominal references

Reduced ML: syntax

Reduced ML: typing rules

Reduced ML: reduction rules

Examples

Contextual equivalence

Examples

Full abstraction

Nominal Games

Strategies

Local state

Considerations

Name innocence

Restrictions on plays

Restrictions on strategies

Name blindness

Results

Conclusion