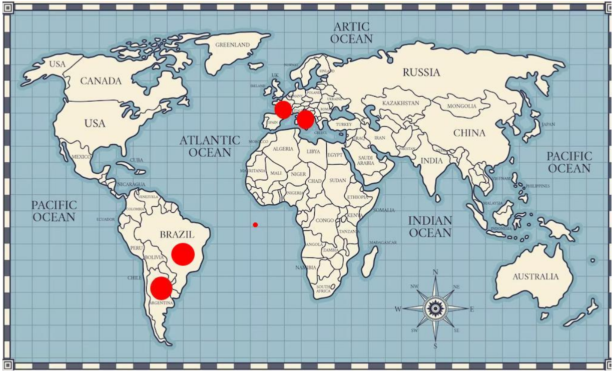
Quantitative Inhabitation

Victor Arrial   Delia Kesner

Joint work with Giulio Guerrieri

60 ans d’Antonio Bucciarelli, Paris, 20 Juin 2023

International Antonio



Italy

France

Argentina

Brazil

Some Fruitful Collaborations on Intersection Types

2016 and 2017 **Non-Idempotent Intersection Types for Lambda-Calculus**.
Bucciarelli-Kesner-Ventura.
🇫🇷 🇧🇷

2014 and 2018 **Inhabitation for Non-Idempotent Intersection Types**.
Bucciarelli-Kesner-RonchiDellaRocca
🇫🇷 🇮🇹

2020 **The Bang Calculus Revisited**.
Bucciarelli-Kesner-Rios-Viso
🇫🇷 🇦🇷

That Inspired an Amazing Paper ...

That Inspired an Amazing Paper ...

Trailer

That Inspired an Amazing Paper ...

Trailer

BANG
NAME
VALUE
OTHERS

60 60 60 60 60

60 60

60 60

60 60

60 60

60 60

60 60 60 60

What is Inhabitation ?

What is Inhabitation ?

Typing Problem:

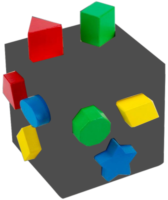
t

What is Inhabitation ?

Typing Problem:

$$\Gamma \vdash t : \sigma$$

What is Inhabitation ?



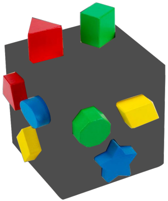
Typing Problem:

$\Gamma \vdash t : \sigma$

Computational: **[Milner78]**

Typers

What is Inhabitation ?



Typing Problem:

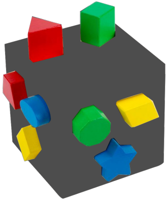
$\Gamma \vdash t : \sigma$

Inhabitation Problem (IP):

Computational: **[Milner78]**

Typers

What is Inhabitation ?

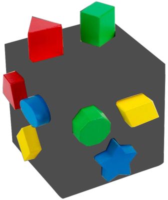


Typing Problem:
$\Gamma \vdash t : \sigma$

Inhabitation Problem (IP):
$\Gamma \qquad \sigma$

Computational: **[Milner78]**
Typers

What is Inhabitation ?



Typing Problem:

$\Gamma \vdash t : \sigma$

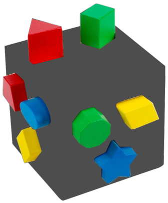
Inhabitation Problem (IP):

$\Gamma \vdash t : \sigma$

Computational: **[Milner78]**

Typers

What is Inhabitation ?

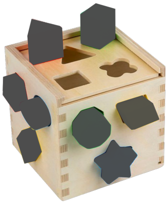


Typing Problem:

$\Gamma \vdash t : \sigma$

Computational: **[Milner78]**

Typers



Inhabitation Problem (IP):

$\Gamma \vdash t : \sigma$

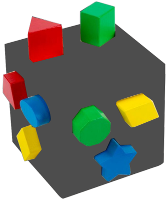
Computational: **[HughesOrchard20]**

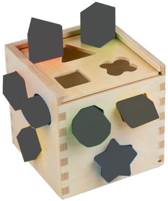
Program Synthesis

Logical: **[HodasMiller94]**

Proof Search and Logic Programming

What is Inhabitation ?





Typing Problem:

$\Gamma \vdash t : \sigma$

Inhabitation Problem (IP):

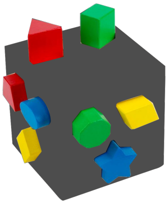
$\Gamma \vdash t : \sigma$

Computational: **[Milner78]**
Typers

Computational: **[HughesOrchard20]**
Program Synthesis

Logical: **[HodasMiller94]**
Proof Search and Logic Programming

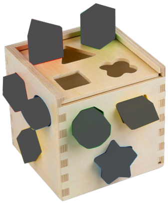
What is Inhabitation ?



Typing Problem:

$\Gamma \vdash t : \sigma$

Computational: **[Milner78]**
Typers



Inhabitation Problem (IP):

$\Gamma \vdash t : \sigma$

Computational: **[HughesOrchard20]**
Program Synthesis

Logical: **[HodasMiller94]**
Proof Search and Logic Programming

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

Unifying Frameworks

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Unifying Frameworks:

- Call-by-Push-Value **[Levy99]**

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Unifying Frameworks:

- Call-by-Push-Value **[Levy99]**
- **Bang Calculus [EhrhardGuerrieri16]**

BANG

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Unifying Frameworks:

- Call-by-Push-Value **[Levy99]**
- **Bang Calculus [EhrhardGuerrieri16]**:

$$t, u \quad ::= \quad x \mid \lambda x.t \mid tu$$

BANG

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Unifying Frameworks:

- Call-by-Push-Value **[Levy99]**
- **Bang Calculus [EhrhardGuerrieri16]**:

$$\begin{array}{rcl} t, u & ::= & x \mid \lambda x.t \mid tu \\ & \mid & !t \end{array}$$

Values

BANG

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Unifying Frameworks:

- Call-by-Push-Value **[Levy99]**
- **Bang Calculus [EhrhardGuerrieri16]**:

$$
\begin{array}{rcll}
t, u & ::= & x \mid \lambda x.t \mid tu & \\
 & \mid & !\, t & \texttt{Values} \\
 & \mid & \texttt{der}(\, t\,) & \texttt{Computations}
\end{array}
$$

BANG

Unifying Frameworks

Different Models of Computation:

Call-by-Name

NAME

Call-by-Value

VALUE

Unifying Frameworks:

- Call-by-Push-Value **[Levy99]**
- **Distant Bang Calculus [EhrhardGuerrieri16] [BucciarelliKesnerRiosViso20,23]**:

$$\begin{array}{rcll} t, u & ::= & x \mid \lambda x.t \mid tu & \\ & \mid & !\,t & \texttt{Values} \\ & \mid & \texttt{der}(\,t\,) & \texttt{Computations} \\ & \mid & t\,[\,x := u\,] & \texttt{Let} \end{array}$$

BANG

Unifying Frameworks

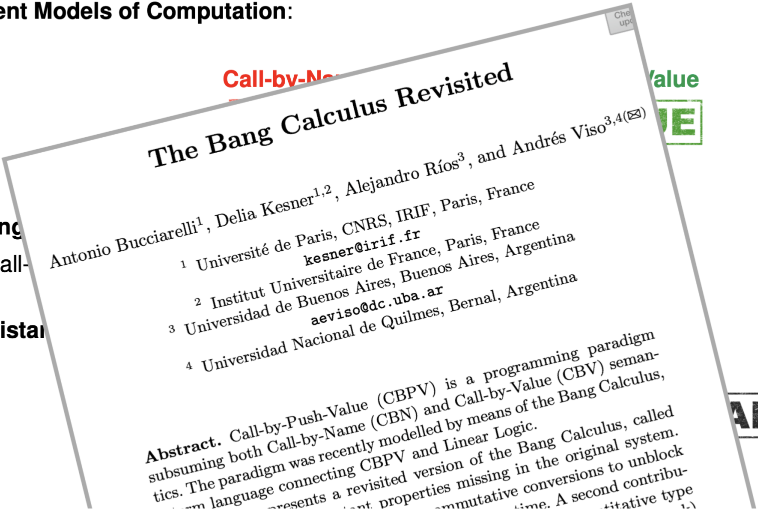
Different Models of Computation:

Call-by-Na

alue

Unifying

- Call-
- **Distar**



The Bang Calculus Revisited

Antonio Bucciarelli[1], Delia Kesner[1,2], Alejandro Ríos[3], and Andrés Viso[3,4](✉)

[1] Université de Paris, CNRS, IRIF, Paris, France
kesner@irif.fr

[2] Institut Universitaire de France, Paris, France

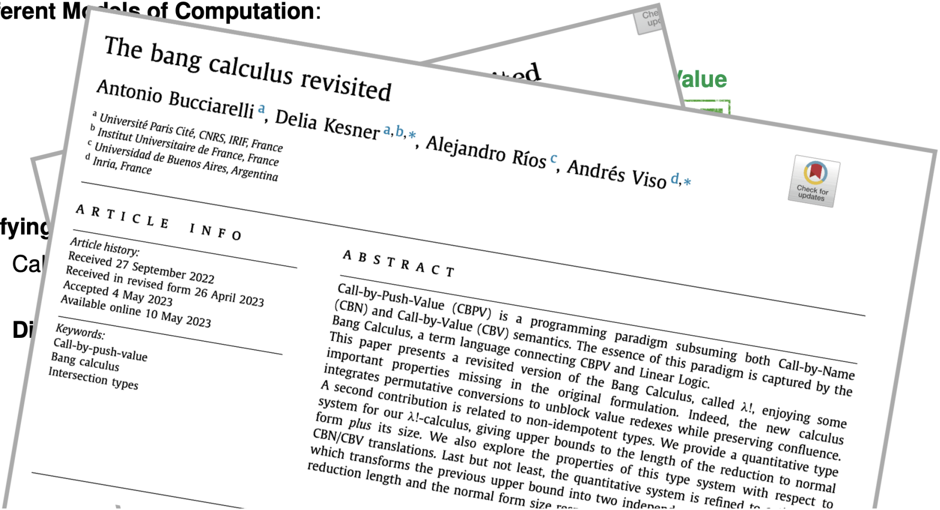
[3] Universidad de Buenos Aires, Buenos Aires, Argentina
aeviso@dc.uba.ar

[4] Universidad Nacional de Quilmes, Bernal, Argentina

Abstract. Call-by-Push-Value (CBPV) is a programming paradigm subsuming both Call-by-Name (CBN) and Call-by-Value (CBV) semantics. The paradigm was recently modelled by means of the Bang Calculus, a
rm language connecting CBPV and Linear Logic.
presents a revisited version of the Bang Calculus, called
ant properties missing in the original system.
mmutative conversions to unblock
time. A second contribu-
titative type

Unifying Frameworks

Different Models of Computation:



The bang calculus revisited

Antonio Bucciarelli [a], Delia Kesner [a,b,*], Alejandro Ríos [c], Andrés Viso [d,*]

[a] *Université Paris Cité, CNRS, IRIF, France*
[b] *Institut Universitaire de France, France*
[c] *Universidad de Buenos Aires, Argentina*
[d] *Inria, France*

ARTICLE INFO

Article history:
Received 27 September 2022
Received in revised form 26 April 2023
Accepted 4 May 2023
Available online 10 May 2023

Keywords:
Call-by-push-value
Bang calculus
Intersection types

ABSTRACT

Call-by-Push-Value (CBPV) is a programming paradigm subsuming both Call-by-Name (CBN) and Call-by-Value (CBV) semantics. The essence of this paradigm is captured by the Bang Calculus, a term language connecting CBPV and Linear Logic.
This paper presents a revisited version of the Bang Calculus, called λ!, enjoying some important properties missing in the original formulation. Indeed, the new calculus integrates permutative conversions to unblock value redexes while preserving confluence.
A second contribution is related to non-idempotent types. We provide a quantitative type system for our λ!-calculus, giving upper bounds to the length of the reduction to normal form *plus* its size. We also explore the properties of this type system with respect to CBN/CBV translations. Last but not least, the quantitative system is refined to which transforms the previous upper bound into two indepen reduction length and the normal form size res

Unifying

- Cal

- **Di**

Distant Bang: A Subsuming Paradigm

Distant Bang: A Subsuming Paradigm

$$t^{N} : \boxed{\text{NAME}} \rightarrow \boxed{\text{BANG}}$$

Distant Bang: A Subsuming Paradigm

t^N : NAME → BANG

Static Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME t normal form

Distant Bang: A Subsuming Paradigm

t^{N} : NAME → BANG

Static Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME t normal form ⇔ t^{N} normal form BANG

Distant Bang: A Subsuming Paradigm

t^{N} : NAME → BANG

Static Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME t normal form ⇔ t^{N} normal form BANG

Dynamic Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME $t \twoheadrightarrow u$

Distant Bang: A Subsuming Paradigm

t^N : NAME → BANG

Static Properties: [BucciarelliKesnerRiosViso20,23, Arrial23]

NAME t normal form ⇔ t^N normal form BANG

Dynamic Properties: [BucciarelliKesnerRiosViso20,23, Arrial23]

NAME $t \twoheadrightarrow u$ ⇔ $t^N \twoheadrightarrow u^N$ BANG

Distant Bang: A Subsuming Paradigm

t^{N} : NAME → BANG

t^{V} : VALUE → BANG

Static Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME $\quad t$ normal form $\Leftrightarrow$ t^{N} normal form $\quad$ BANG

Dynamic Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME $\quad t \twoheadrightarrow u \quad \Leftrightarrow \quad t^{N} \twoheadrightarrow u^{N}$ $\quad$ BANG

Distant Bang: A Subsuming Paradigm

t^{N} : NAME $\to$ BANG

t^{V} : VALUE $\to$ BANG

Static Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME t normal form $\Leftrightarrow$ t^{N} normal form

VALUE t normal form $\Leftrightarrow$ t^{V} normal form

BANG

Dynamic Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME $t \twoheadrightarrow u \Leftrightarrow t^{N} \twoheadrightarrow u^{N}$

VALUE $t \twoheadrightarrow u \Leftrightarrow t^{V} \twoheadrightarrow u^{V}$

BANG

Distant Bang: A Subsuming Paradigm

t^N : NAME → BANG

t^V : VALUE → BANG

Static Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME t normal form ⇔ t^N normal form

VALUE t normal form ⇔ t^V normal form

BANG

Dynamic Properties: **[BucciarelliKesnerRiosViso20,23, Arrial23]**

NAME $t \twoheadrightarrow u$ ⇔ $t^N \twoheadrightarrow u^N$

VALUE $t \twoheadrightarrow u$ ⇔ $t^V \twoheadrightarrow u^V$

BANG

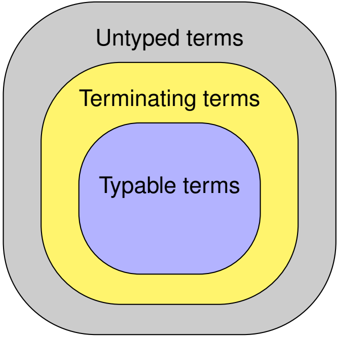
Can we do the same thing with inhabitation ?

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

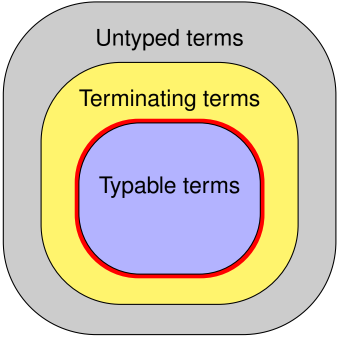
Simple Types Versus Intersection Types

$A, B ::= \sigma \mid A \Rightarrow B$



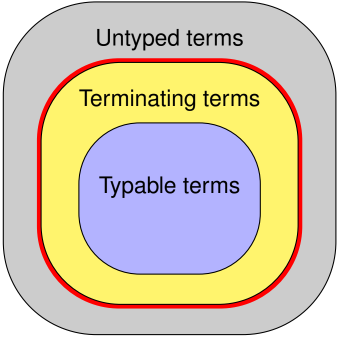
Simple Types Versus Intersection Types

$A, B ::= \sigma \mid A \Rightarrow B$



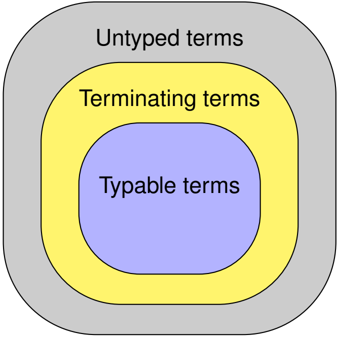
Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B$$



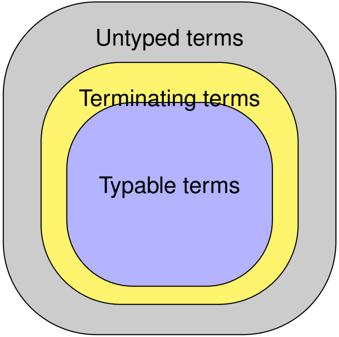
Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



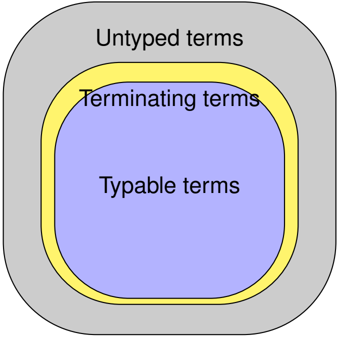
Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



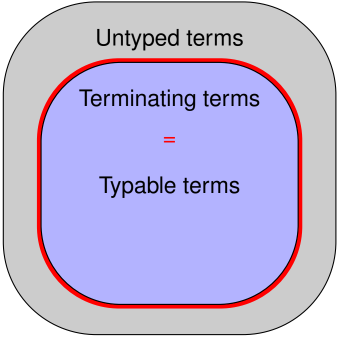
Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



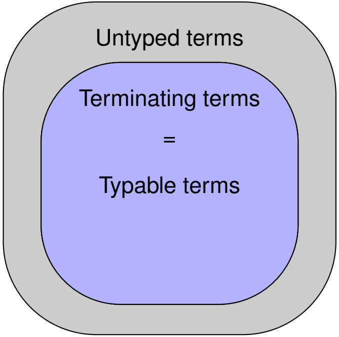
Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



Untyped terms

Terminating terms

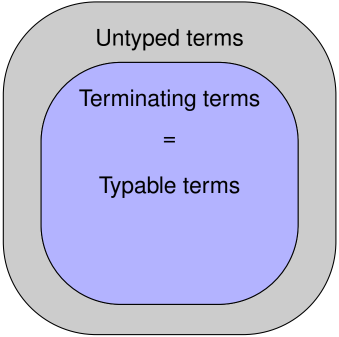
=

Typable terms

- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$

Simple Types Versus Intersection Types

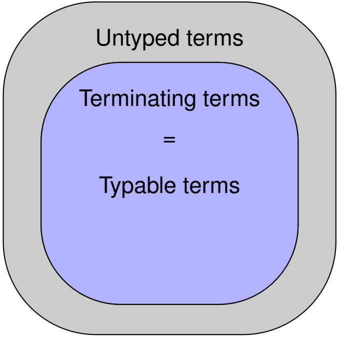
$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$
- **Commutativity**:
  $A \cap B = B \cap A$

Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid {\color{red} A \cap B}$$



Untyped terms

Terminating terms

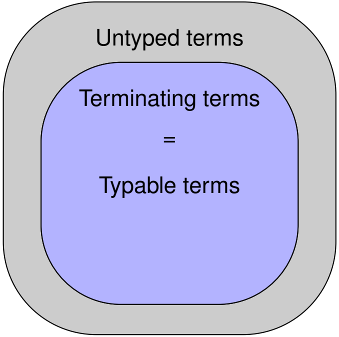
=

Typable terms

- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$
- **Commutativity**:
  $A \cap B = B \cap A$
- **Idempotency**?

Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



Untyped terms

Terminating terms

=

Typable terms

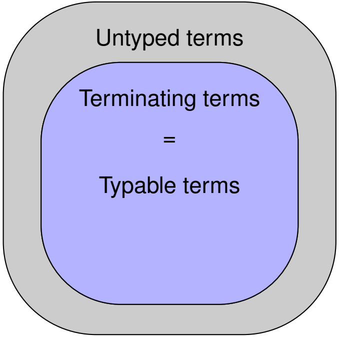
- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$
- **Commutativity**:
  $A \cap B = B \cap A$
- **Idempotency**?

Idempotent
[CoppoDezani78,80]

$A \cap A = A$

Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



Untyped terms

Terminating terms

=

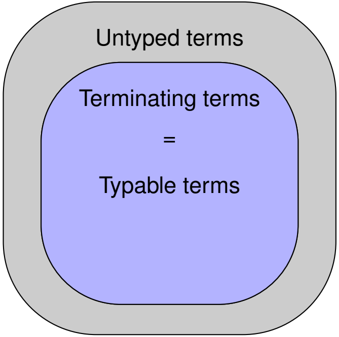
Typable terms

- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$
- **Commutativity**:
  $A \cap B = B \cap A$
- **Idempotency**?

Idempotent [CoppoDezani78,80] $A \cap A = A$	
<span style="color:red">Qualitative</span> properties ✓ ✗	

Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



Untyped terms

Terminating terms

=

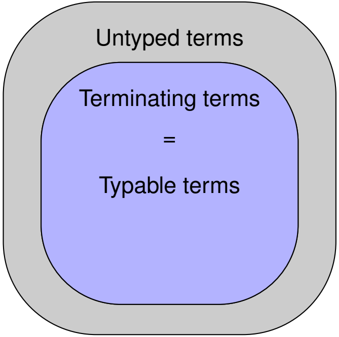
Typable terms

- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$
- **Commutativity**:
  $A \cap B = B \cap A$
- **Idempotency**?

Idempotent **[CoppoDezani78,80]**	**Non-Idempotent** **[Gardner94]**, **[Kfoury00]**
$A \cap A = A$	$A \cap A \neq A$
Qualitative properties ✓ ✗	

Simple Types Versus Intersection Types

$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$



Untyped terms

Terminating terms

=

Typable terms

- **Associativity**:
  $A \cap (B \cap C) = (A \cap B) \cap C$
- **Commutativity**:
  $A \cap B = B \cap A$
- **Idempotency**?

Idempotent **[CoppoDezani78,80]** $A \cap A = A$	**Non-Idempotent** **[Gardner94]**, **[Kfoury00]** $A \cap A \neq A$
Qualitative properties ✓ ✗	Quantitative properties **[deCarvalho07]**

Simple Types Versus Intersection Types

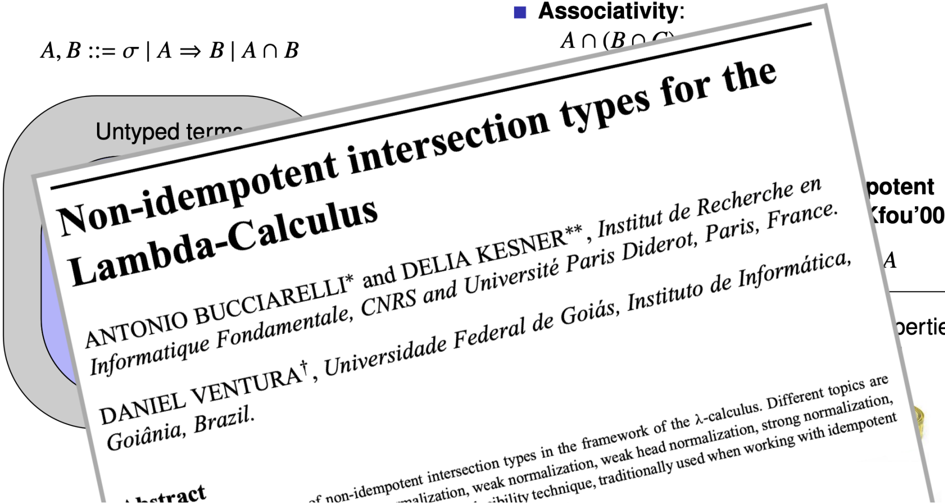
$$A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B$$

- **Associativity**:
  $A \cap (B \cap C)$

potent
Kfou'00]

A

perties



Non-idempotent intersection types for the Lambda-Calculus

ANTONIO BUCCIARELLI* and DELIA KESNER**, *Institut de Recherche en Informatique Fondamentale, CNRS and Université Paris Diderot, Paris, France.*

DANIEL VENTURA†, *Universidade Federal de Goiás, Instituto de Informática, Goiânia, Brazil.*

Abstract

of non-idempotent intersection types in the framework of the λ-calculus. Different topics are
malization, weak normalization, weak head normalization, strong normalization,
ibility technique, traditionally used when working with idempotent

Typability and Inhabitation in Intersection Types

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : \ ?$	**Inhabitation** $\Gamma \vdash \ ? : \sigma$

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types		
Idempotent Types		
Non-Idempotent Types		

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t :\ ?$	**Inhabitation** $\Gamma \vdash\ ?\ : \sigma$
Simple Types	Decidable	
Idempotent Types		
Non-Idempotent Types		

Typability and Inhabitation in Intersection Types

	Typing	**Inhabitation**
	$? \vdash t : ?$	$\Gamma \vdash ? : \sigma$
Simple Types	Decidable	
Idempotent Types	Indecidable	
Non-Idempotent Types	Indecidable	

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	
Non-Idempotent Types	Indecidable	

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	Indecidable **[Urzyczyn99]**
Non-Idempotent Types	Indecidable	

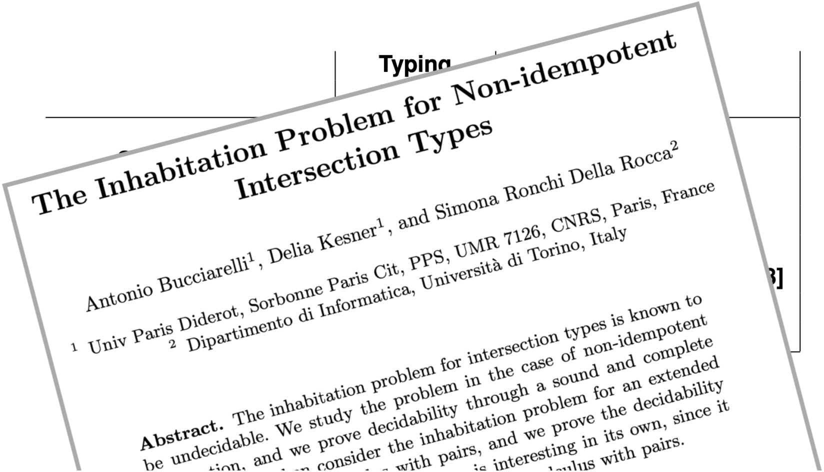
Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	Indecidable **[Urzyczyn99]**
Non-Idempotent Types	Indecidable	(CBN) Decidable **[BKR’18]**

Typability and Inhabitation in Intersection Types

Typing

3]



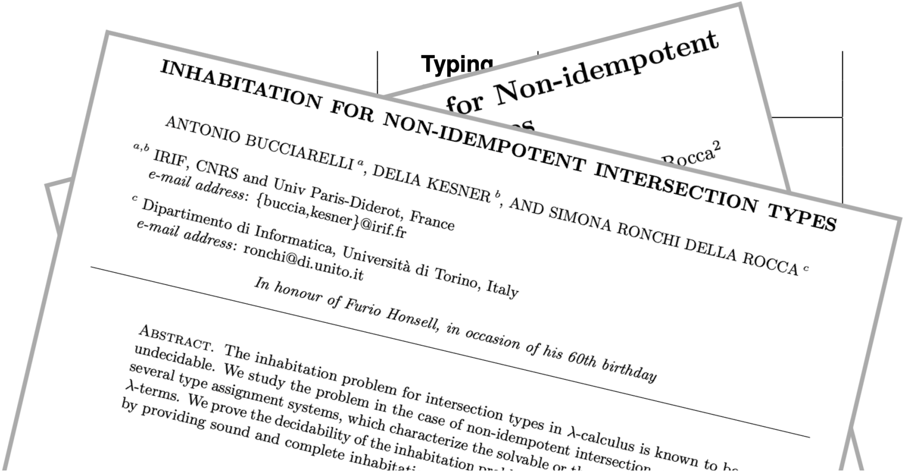
The Inhabitation Problem for Non-idempotent Intersection Types

Antonio Bucciarelli[1], Delia Kesner[1], and Simona Ronchi Della Rocca[2]

[1] Univ Paris Diderot, Sorbonne Paris Cit, PPS, UMR 7126, CNRS, Paris, France
[2] Dipartimento di Informatica, Università di Torino, Italy

Abstract. The inhabitation problem for intersection types is known to be undecidable. We study the problem in the case of non-idempotent tion, and we prove decidability through a sound and complete n consider the inhabitation problem for an extended s with pairs, and we prove the decidability is interesting in its own, since it lculus with pairs.

Typability and Inhabitation in Intersection Types



Typing

for Non-idempotent

Rocca^2

INHABITATION FOR NON-IDEMPOTENT INTERSECTION TYPES

ANTONIO BUCCIARELLI [a], DELIA KESNER [b], AND SIMONA RONCHI DELLA ROCCA [c]

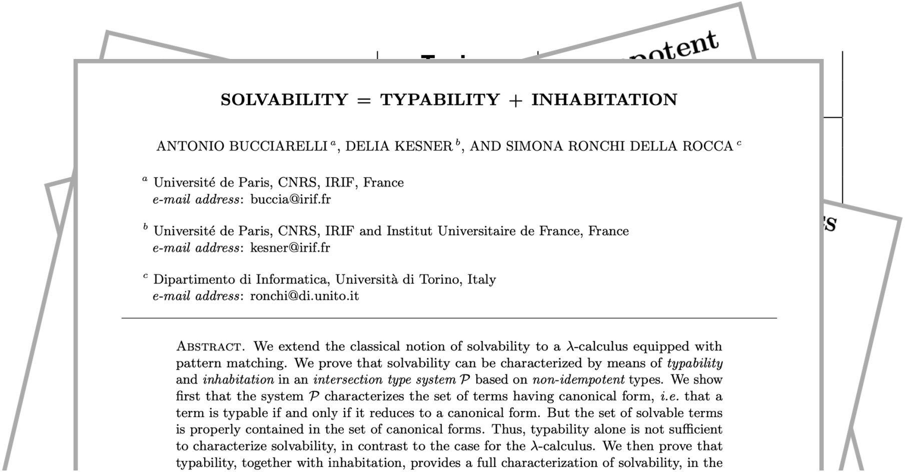
[a,b] IRIF, CNRS and Univ Paris-Diderot, France
e-mail address: {buccia,kesner}@irif.fr

[c] Dipartimento di Informatica, Università di Torino, Italy
e-mail address: ronchi@di.unito.it

In honour of Furio Honsell, in occasion of his 60th birthday

Abstract. The inhabitation problem for intersection types in λ-calculus is known to be undecidable. We study the problem in the case of non-idempotent intersection
several type assignment systems, which characterize the solvable or th
λ-terms. We prove the decidability of the inhabitation pro
by providing sound and complete inhabitati

Typability and Inhabitation in Intersection Types



SOLVABILITY = TYPABILITY + INHABITATION

ANTONIO BUCCIARELLI[a], DELIA KESNER[b], AND SIMONA RONCHI DELLA ROCCA[c]

[a] Université de Paris, CNRS, IRIF, France
e-mail address: buccia@irif.fr

[b] Université de Paris, CNRS, IRIF and Institut Universitaire de France, France
e-mail address: kesner@irif.fr

[c] Dipartimento di Informatica, Università di Torino, Italy
e-mail address: ronchi@di.unito.it

Abstract. We extend the classical notion of solvability to a λ-calculus equipped with pattern matching. We prove that solvability can be characterized by means of *typability* and *inhabitation* in an *intersection type system* $\mathcal{P}$ based on *non-idempotent* types. We show first that the system $\mathcal{P}$ characterizes the set of terms having canonical form, *i.e.* that a term is typable if and only if it reduces to a canonical form. But the set of solvable terms is properly contained in the set of canonical forms. Thus, typability alone is not sufficient to characterize solvability, in contrast to the case for the λ-calculus. We then prove that typability, together with inhabitation, provides a full characterization of solvability, in the

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	Indecidable **[Urzyczyn99]**
Non-Idempotent Types	Indecidable	(CBN) Decidable **[BKR'18]**

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	Indecidable **[Urzyczyn99]**
Non-Idempotent Types	Indecidable	(CBN) Decidable **[BKR’18]** (CBV) ?

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	Indecidable **[Urzyczyn99]**
Non-Idempotent Types	Indecidable	(CBN) Decidable **[BKR'18]** (CBV) ?

Typability and Inhabitation in Intersection Types

	Typing $? \vdash t : ?$	**Inhabitation** $\Gamma \vdash ? : \sigma$
Simple Types	Decidable	Decidable
Idempotent Types	Indecidable	Indecidable **[Urzyczyn99]**
Non-Idempotent Types	Indecidable	(CBN) Decidable **[BKR'18]** (CBV) Decidable

Intersection Types and Distant Bang Calculus

Intersection Types and Distant Bang Calculus

Three Typing Systems: [BucciarelliKesnerRiosViso20,23]

NAME : $\mathcal{N}$ VALUE : $\mathcal{V}$ BANG : $\mathcal{B}$

Intersection Types and Distant Bang Calculus

Three Typing Systems: **[BucciarelliKesnerRiosViso20,23]**

NAME : $\mathcal{N}$ VALUE : $\mathcal{V}$ BANG : $\mathcal{B}$

Static Properties: **[BucciarelliKesnerRiosViso20,23]**

NAME $\Gamma \vdash_{\mathcal{N}} t : \sigma$

Intersection Types and Distant Bang Calculus

Three Typing Systems: **[BucciarelliKesnerRiosViso20,23]**

NAME : $\mathcal{N}$ VALUE : $\mathcal{V}$ BANG : $\mathcal{B}$

Static Properties: **[BucciarelliKesnerRiosViso20,23]**

NAME $\Gamma \vdash_{\mathcal{N}} t : \sigma \quad \Leftrightarrow \quad \Gamma \vdash_{\mathcal{B}} t^{N} : \sigma$ BANG

Intersection Types and Distant Bang Calculus

Three Typing Systems: **[BucciarelliKesnerRiosViso20,23]**

NAME : $\mathcal{N}$ VALUE : $\mathcal{V}$ BANG : $\mathcal{B}$

Static Properties: **[BucciarelliKesnerRiosViso20,23]**

NAME $\Gamma \vdash_{\mathcal{N}} t : \sigma \quad \Leftrightarrow \quad \Gamma \vdash_{\mathcal{B}} t^{N} : \sigma$

VALUE $\Gamma \vdash_{\mathcal{V}} t : \sigma \quad \Leftrightarrow \quad \Gamma \vdash_{\mathcal{B}} t^{V} : \sigma$

BANG

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework

Coming Back to Inhabitation

Coming Back to Inhabitation

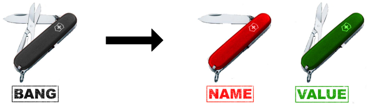
First Goal

- **Decidability** of the (more general) BANG Inhabitation Problem (IP).

Coming Back to Inhabitation

First Goal + More Ambitious Second Goal

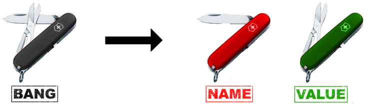
- **Decidability** of the (more general) BANG Inhabitation Problem (IP).
- **Decidability** of the NAME and VALUE IP from **decidability** of the BANG IP.



Coming Back to Inhabitation

First Goal + More Ambitious Second Goal

- **Decidability** of the (more general) BANG Inhabitation Problem (IP).
- **Decidability** of the NAME and VALUE IP from **decidability** of the BANG IP.

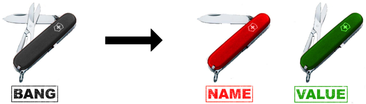


More Ambitious Third Goal

Coming Back to Inhabitation

First Goal + More Ambitious Second Goal

- **Decidability** of the (more general) BANG Inhabitation Problem (IP).
- **Decidability** of the NAME and VALUE IP from **decidability** of the BANG IP.



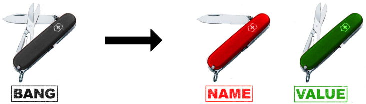
More Ambitious Third Goal

- Decidability by **finding all inhabitants** in the BANG IP.

Coming Back to Inhabitation

First Goal + More Ambitious Second Goal

- **Decidability** of the (more general) BANG Inhabitation Problem (IP).
- **Decidability** of the NAME and VALUE IP from **decidability** of the BANG IP.



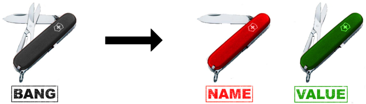
More Ambitious Third Goal

- Decidability by **finding all inhabitants** in the BANG IP.
- Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.

Coming Back to Inhabitation

First Goal + More Ambitious Second Goal

- **Decidability** of the (more general) BANG Inhabitation Problem (IP).
- **Decidability** of the NAME and VALUE IP from **decidability** of the BANG IP.



More Ambitious Third Goal

- Decidability by **finding all inhabitants** in the BANG IP.
- Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Solving the Inhabitation Problem - Methodology

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\texttt{Sol}(\Gamma, \sigma) \;:=\; \{\, \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \,\}$$

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\texttt{Sol}(\Gamma, \sigma) \ := \ \{ \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \}$$

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\texttt{Sol}(\Gamma, \sigma) \;:=\; \{\, \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \,\}$$

Problem

☒ The set $\texttt{Sol}(\Gamma, \sigma)$ is either empty of infinite BANG

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\mathtt{Sol}(\Gamma,\sigma) \;:=\; \{\, \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \,\}$$

Problem

- The set $\mathtt{Sol}(\Gamma,\sigma)$ is either empty of infinite BANG

We compute a **finite** generator:

$$\mathtt{Basis}(\Gamma,\sigma)$$

Which is **correct** and **complete**:

$$\mathtt{span}(\,\mathtt{Basis}(\Gamma,\sigma)\,) \;=\; \mathtt{Sol}(\Gamma,\sigma)$$

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\mathtt{Sol}(\Gamma, \sigma) \ := \ \{\, \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \,\}$$

Problem

✗ The set $\mathtt{Sol}(\Gamma, \sigma)$ is either empty of infinite **BANG**

We compute a **finite** generator:

$$\mathtt{Basis}(\Gamma, \sigma)$$

Which is **correct** and **complete**:

$$\mathtt{span}(\,\mathtt{Basis}(\Gamma, \sigma)\,) \ = \ \mathtt{Sol}(\Gamma, \sigma)$$

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\mathtt{Sol}(\Gamma, \sigma) \;:=\; \{\, \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \,\}$$

Problem

✗ The set $\mathtt{Sol}(\Gamma, \sigma)$ is either empty of infinite

BANG

We compute a **finite** generator:

$$\mathtt{Basis}(\Gamma, \sigma)$$

Which is **correct** and **complete**:

$$\mathtt{span}(\mathtt{Basis}(\Gamma, \sigma)) \;=\; \mathtt{Sol}(\Gamma, \sigma)$$

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\mathtt{Sol}(\Gamma,\sigma) \; := \; \{\, \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \,\}$$

Problem

✗ The set $\mathtt{Sol}(\Gamma,\sigma)$ is either empty of infinite **BANG**

We compute a **finite** generator:

$$\mathtt{Basis}(\Gamma,\sigma)$$

Which is **correct** and **complete**:

$$\mathtt{span}(\,\mathtt{Basis}(\Gamma,\sigma)\,) \; = \; \mathtt{Sol}(\Gamma,\sigma)$$

Theorem

✓ *For any typing* (Γ,σ)*,* $\mathtt{Basis}_{\mathcal{B}}(\Gamma,\sigma)$ ***exists****, is* ***finite****,* ***correct*** *and* ***complete****.* **BANG**

Solving the Inhabitation Problem - Methodology

Instead of **just one** solution:

$$\Gamma \vdash \mathbf{t} : \sigma$$

We want to compute **all** solutions:

$$\mathtt{Sol}(\Gamma, \sigma) \ := \ \{ \mathbf{t} \mid \Gamma \vdash \mathbf{t} : \sigma \}$$

Problem

✗ The set $\mathtt{Sol}(\Gamma, \sigma)$ is either empty of infinite **BANG**

We compute a **finite** generator:

$$\mathtt{Basis}(\Gamma, \sigma)$$

Which is **correct** and **complete**:

$$\mathtt{span}(\, \mathtt{Basis}(\Gamma, \sigma) \,) \ = \ \mathtt{Sol}(\Gamma, \sigma)$$

Theorem

✓ *For any typing* (Γ, σ), $\mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ ***exists***, *is* ***finite***, ***correct*** *and* ***complete***. **BANG**

Following the Typing and a Grammar

Following the Typing and a Grammar

Computing the basis:

Recreate typing trees, but only on elements of the `Basis`.

Following the Typing and a Grammar

Computing the basis:

Recreate typing trees, but only on elements of the `Basis`.

Follows two sets of rules:

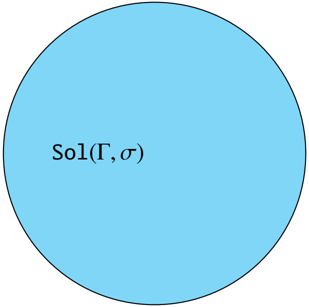
Following the Typing and a Grammar

Computing the basis:

Recreate typing trees, but only on elements of the `Basis`.

Follows two sets of rules:

- Typing rules



$\mathtt{Sol}(\Gamma, \sigma)$

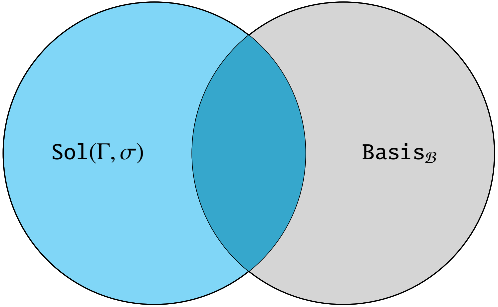
Following the Typing and a Grammar

Computing the basis:

Recreate typing trees, but only on elements of the `Basis`.

Follows two sets of rules:

- Typing rules
- Grammar rules


$\mathtt{Sol}(\Gamma, \sigma)$
$\mathtt{Basis}_{\mathcal{B}}$

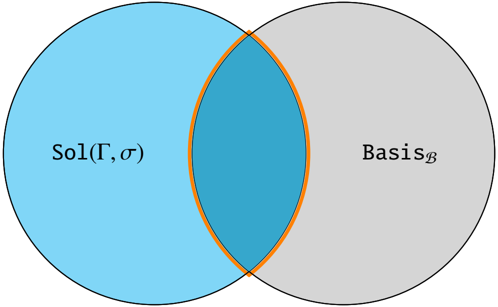
Following the Typing and a Grammar

Computing the basis:

Recreate typing trees, but only on elements of the `Basis`.

Follows two sets of rules:

- Typing rules
- Grammar rules


$\texttt{Sol}(\Gamma, \sigma)$
$\texttt{Basis}_{\mathcal{B}}$

The Full Algorithm

$$\frac{g \rightsquigarrow \mathsf{Var} \mid}{x \Vdash_g H^{x:[\sigma]}(\emptyset;\sigma)}\,\text{VAR} \qquad \frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau,\Diamond) \end{array} & a \Vdash_{g'} H^{x:[\tau]}(\Gamma;[\sigma]) \end{array}}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{DR}$$

$$\frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{App}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau,\Diamond \Rightarrow \sigma) \end{array} & a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a;\mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b;\mathcal{M}) \end{array}}{ab \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{APP}$$

$$\frac{g \rightsquigarrow g' \mid \ a \Vdash_{g'} H^{x:[\tau]}(\Gamma;\sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{H-H} \qquad \frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow g' \\ \Gamma = \Gamma' + x : [\tau] \\ \sigma \Vvdash S(\tau,\Diamond) \end{array} & a \Vdash_{g'} H^{x:[\tau]}(\Gamma';\sigma) \end{array}}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-H} \qquad \frac{g \rightsquigarrow g' \mid \ a \Vdash_{g'} N(\Gamma;\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-N}$$

$$\frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{Lam}(g') \\ \mathtt{fix}\ \ x \notin \mathsf{dom}(\Gamma) \end{array} & a \Vdash_{g'} N(\Gamma, x : \mathcal{M};\sigma) \end{array}}{\lambda x.a \Vdash_g N(\Gamma;\mathcal{M} \Rightarrow \sigma)}\,\text{ABS} \qquad \frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{Bng}(g') \\ I \neq \emptyset \\ \Gamma = +_{i \in I}\Gamma_i \end{array} & \left(a_i \Vdash_{g'} N(\Gamma_i;\tau_i)\right)_{i \in I} \quad \uparrow_{i \in I} a_i \end{array}}{!\bigvee_{i \in I} a_i \Vdash_g N(\Gamma;[\tau_i]_{i \in I})}\,\text{BG} \qquad \frac{g \rightsquigarrow \mathsf{Bng}(\bot) \mid}{!\bot \Vdash_g N(\emptyset;[\,])}\,\text{BG}_\bot$$

$$\frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\rho], \qquad \mathtt{fix}\ \ y \notin \mathsf{dom}(\Gamma) \cup \{x\} \\ n \in \llbracket 0, \mathsf{sz}(\rho) \rrbracket,\ \mathcal{M} \Vvdash S(\rho,[\Diamond_1,\ldots,\Diamond_n]) \end{array} & a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y{:}\,\mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\rho]}(\Gamma_b;\mathcal{M}) \end{array}}{a[y \backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-H}$$

$$\frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \mathtt{fix}\ \ y \notin \mathsf{dom}(\Gamma) \cup \{x\} \\ n \in \llbracket 1, \mathsf{sz}(\tau) \rrbracket,\ [\rho_i]_{i \in \llbracket 1,n \rrbracket} \Vvdash S(\tau,[\Diamond_1,\ldots,\Diamond_n]) \\ j \in \llbracket 1,n \rrbracket,\ \sigma \Vvdash S(\rho_j,\Diamond) \end{array} & a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y : [\rho_i]_{i \in \llbracket 1,n \rrbracket \backslash j};\sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b;[\rho_i]_{i \in \llbracket 1,n \rrbracket}) \end{array}}{a[y \backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-CH}$$

$$\frac{\begin{array}{c|c} \begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\tau], \qquad \mathtt{fix}\ \ y \notin \mathsf{dom}(\Gamma) \\ n \in \llbracket 0, \mathsf{sz}(\tau) \rrbracket,\ \mathcal{M} \Vvdash S(\tau,[\Diamond_1,\ldots,\Diamond_n]) \end{array} & a \Vdash_{g_a} N(\Gamma_a, y : \mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b;\mathcal{M}) \end{array}}{a[y \backslash b] \Vdash_g N(\Gamma;\sigma)}\,\text{ES-N}$$

The Full Algorithm

$$\frac{g \rightsquigarrow \mathsf{Var} \;\mid}{x \Vdash_g H^{x:[\sigma]}(\emptyset;\sigma)}\,\text{VAR} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau,\diamond) \end{array} \;\Bigg|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;[\sigma])}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{DR}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{App}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau,\diamond \Rightarrow \sigma) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a;\mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b;\mathcal{M})}{ab \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{APP}$$

$$\frac{g \rightsquigarrow g' \;\mid\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;\sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{H-H} \qquad \frac{\begin{array}{c} g \rightsquigarrow g' \\ \Gamma = \Gamma' + x:[\tau] \\ \sigma \Vvdash S(\tau,\diamond) \end{array} \;\Bigg|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma';\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-H} \qquad \frac{g \rightsquigarrow g' \;\mid\; a \Vdash_{g'} N(\Gamma;\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-N}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Lam}(g') \\ \mathtt{fix}\;\; x \notin \mathsf{dom}(\Gamma) \end{array} \;\Bigg|\; a \Vdash_{g'} N(\Gamma, x:\mathcal{M};\sigma)}{\lambda x.a \Vdash_g N(\Gamma;\mathcal{M} \Rightarrow \sigma)}\,\text{ABS} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Bng}(g') \\ I \neq \emptyset \\ \Gamma = +_{i\in I}\Gamma_i \end{array} \;\Bigg|\; \left(a_i \Vdash_{g'} N(\Gamma_i;\tau_i)\right)_{i\in I} \quad \uparrow_{i\in I} a_i}{!\bigvee_{i\in I} a_i \Vdash_g N(\Gamma;[\tau_i]_{i\in I})}\,\text{BG} \qquad \frac{g \rightsquigarrow \mathsf{Bng}(\bot) \;\mid}{!\bot \Vdash_g N(\emptyset;[\,])}\,\text{BG}_\bot$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z:[\rho], \qquad \mathtt{fix}\;\; y \notin \mathsf{dom}(\Gamma) \cup \{x\} \\ n \in [\![0,\mathsf{sz}(\rho)]\!],\; \mathcal{M} \Vvdash S(\rho,[\diamond_1,\ldots,\diamond_n]) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y:\mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\rho]}(\Gamma_b;\mathcal{M})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-H}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \mathtt{fix}\;\; y \notin \mathsf{dom}(\Gamma) \cup \{x\} \\ n \in [\![1,\mathsf{sz}(\tau)]\!],\; [\rho_i]_{i\in[\![1,n]\!]} \Vvdash S(\tau,[\diamond_1,\ldots,\diamond_n]) \\ j \in [\![1,n]\!],\; \sigma \Vvdash S(\rho_j,\diamond) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y:[\rho_i]_{i\in[\![1,n]\!]\backslash j};\sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b;[\rho_i]_{i\in[\![1,n]\!]})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-CH}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z:[\tau], \qquad \mathtt{fix}\;\; y \notin \mathsf{dom}(\Gamma) \\ n \in [\![0,\mathsf{sz}(\tau)]\!],\; \mathcal{M} \Vvdash S(\tau,[\diamond_1,\ldots,\diamond_n]) \end{array} \;\Bigg|\; a \Vdash_{g_a} N(\Gamma_a, y:\mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b;\mathcal{M})}{a[y\backslash b] \Vdash_g N(\Gamma;\sigma)}\,\text{ES-N}$$

The Full Algorithm

$$\frac{g \rightsquigarrow \mathsf{Var} \;\Big|}{x \Vdash_g H^{x:[\sigma]}(\emptyset;\sigma)}\,\text{VAR} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau,\Diamond) \end{array} \;\Bigg|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;[\sigma])}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{DR}$$

$$\frac{g \rightsquigarrow g' \;\Big|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;\sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{H-H} \qquad \frac{\begin{array}{c} g \rightsquigarrow g' \\ \Gamma = \Gamma' + x : [\tau] \\ \sigma \Vvdash S(\tau,\Diamond) \end{array} \;\Bigg|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma';\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-H} \qquad \frac{g \rightsquigarrow g' \;\Big|\; a \Vdash_{g'} N(\Gamma;\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-N}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{App}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau, \Diamond \Rightarrow \sigma) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a; \mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b; \mathcal{M})}{ab \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{APP}$$

$$\frac{n \in [\![0, \mathrm{sz}(\rho)]\!],\ \mathcal{M} \Vvdash S(\rho, [\Diamond_1, \ldots, \Diamond_n]) \;\Big|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y : \mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\rho]}(\Gamma_b; \mathcal{M})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-H}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \mathtt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in [\![1, \mathrm{sz}(\tau)]\!],\ [\rho_i]_{i \in [\![1,n]\!]} \Vvdash S(\tau, [\Diamond_1, \ldots, \Diamond_n]) \\ j \in [\![1, n]\!],\ \sigma \Vvdash S(\rho_j, \Diamond) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y : [\rho_i]_{i \in [\![1,n]\!] \backslash j};\sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b; [\rho_i]_{i \in [\![1,n]\!]})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-CH}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\tau], \qquad \mathtt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \\ n \in [\![0, \mathrm{sz}(\tau)]\!],\ \mathcal{M} \Vvdash S(\tau, [\Diamond_1, \ldots, \Diamond_n]) \end{array} \;\Bigg|\; a \Vdash_{g_a} N(\Gamma_a, y : \mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b; \mathcal{M})}{a[y\backslash b] \Vdash_g N(\Gamma;\sigma)}\,\text{ES-N}$$

The Full Algorithm

$$\frac{g \rightsquigarrow \mathsf{Var} \;\mid}{x \Vdash_g H^{x:[\sigma]}(\emptyset;\sigma)}\,\text{VAR} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau,\Diamond) \end{array} \;\Big|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;[\sigma])}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{DR}$$

$$\frac{g \rightsquigarrow g' \;\mid\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;\sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{H-H} \qquad \frac{\begin{array}{c} g \rightsquigarrow g' \\ \Gamma = \Gamma' + x:[\tau] \\ \sigma \Vvdash S(\tau,\Diamond) \end{array} \;\Big|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma';\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-H} \qquad \frac{g \rightsquigarrow g' \;\mid\; a \Vdash_{g'} N(\Gamma;\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-N}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{App}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau, \Diamond \Rightarrow \sigma) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a;\mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b;\mathcal{M})}{ab \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{APP}$$

$$\frac{n \in [\![0,\mathrm{sz}(\rho)]\!],\; \mathcal{M} \Vvdash S(\rho,[\Diamond_1,\ldots,\Diamond_n]) \;\mid\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y:\mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{x:[\rho]}(\Gamma_b;\mathcal{M})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-H}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \mathtt{fix}\;\; y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in [\![1,\mathrm{sz}(\tau)]\!],\; [\rho_i]_{i\in[\![1,n]\!]} \Vvdash S(\tau,[\Diamond_1,\ldots,\Diamond_n]) \\ j \in [\![1,n]\!],\; \sigma \Vvdash S(\rho_j,\Diamond) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y:[\rho_i]_{i\in[\![1,n]\!]\backslash j};\sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b;[\rho_i]_{i\in[\![1,n]\!]})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-CH}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a,g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z:[\tau], \qquad \mathtt{fix}\;\; y \notin \mathrm{dom}(\Gamma) \\ n \in [\![0,\mathrm{sz}(\tau)]\!],\; \mathcal{M} \Vvdash S(\tau,[\Diamond_1,\ldots,\Diamond_n]) \end{array} \;\Bigg|\; a \Vdash_{g_a} N(\Gamma_a, y:\mathcal{M};\sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b;\mathcal{M})}{a[y\backslash b] \Vdash_g N(\Gamma;\sigma)}\,\text{ES-N}$$

The Full Algorithm

$$\frac{g \leadsto \mathsf{Var} \quad \mid}{x \Vdash_g H^{x:[\sigma]}(\emptyset;\sigma)}\,\text{VAR}$$

$$\frac{\begin{array}{c} g \leadsto \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau,\diamond) \end{array} \;\Big|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;[\sigma])}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{DR}$$

$$\frac{g \leadsto g' \;\mid\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma;\sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{H-H}$$

$$\frac{\begin{array}{c} g \leadsto g' \\ \Gamma = \Gamma' + x : [\tau] \\ \sigma \Vvdash S(\tau,\diamond) \end{array} \;\Big|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma';\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-H}$$

$$\frac{g \leadsto g' \;\mid\; a \Vdash_{g'} N(\Gamma;\sigma)}{a \Vdash_g N(\Gamma;\sigma)}\,\text{N-N}$$

$$\frac{\begin{array}{c} g \leadsto \mathsf{App}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau, \diamond \Rightarrow \sigma) \end{array} \;\Big|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a; \mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b; \mathcal{M})}{ab \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{APP}$$

$$\frac{n \in [\![0, \mathrm{sz}(\rho)]\!],\ \mathcal{M} \Vvdash S(\rho, [\diamond_1, \ldots, \diamond_n]) \;\mid\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y : \mathcal{M}; \sigma) \quad b \Vdash_{g_b} H^{x:[\rho]}(\Gamma_b; \mathcal{M})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-H}$$

$$\frac{\begin{array}{c} g \leadsto \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \texttt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in [\![1, \mathrm{sz}(\tau)]\!],\ [\rho_i]_{i \in [\![1,n]\!]} \Vvdash S(\tau, [\diamond_1, \ldots, \diamond_n]) \\ j \in [\![1, n]\!],\ \sigma \Vvdash S(\rho_j, \diamond) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y : [\rho_i]_{i \in [\![1,n]\!] \backslash j}; \sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b; [\rho_i]_{i \in [\![1,n]\!]})}{a[y\backslash b] \Vdash_g H^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-CH}$$

$$\frac{\begin{array}{c} g \leadsto \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\tau], \qquad \texttt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \\ n \in [\![0, \mathrm{sz}(\tau)]\!],\ \mathcal{M} \Vvdash S(\tau, [\diamond_1, \ldots, \diamond_n]) \end{array} \;\Bigg|\; a \Vdash_{g_a} N(\Gamma_a, y : \mathcal{M}; \sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b; \mathcal{M})}{a[y\backslash b] \Vdash_g N(\Gamma;\sigma)}\,\text{ES-N}$$

The Full Algorithm

$$\frac{g \rightsquigarrow \mathsf{Var} \quad |}{x \Vdash_g H^{x:[\sigma]}(\emptyset; \sigma)}\,\text{VAR} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau, \diamond) \end{array} \;\Bigg|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma; [\sigma])}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\text{DR}$$

$$\frac{g \rightsquigarrow g' \;\;|\;\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma; \sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\text{H-H} \qquad \frac{\begin{array}{c} g \rightsquigarrow g' \\ \Gamma = \Gamma' + x : [\tau] \\ \sigma \Vvdash S(\tau, \diamond) \end{array} \;\Bigg|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma'; \sigma)}{a \Vdash_g N(\Gamma; \sigma)}\,\text{N-H} \qquad \frac{g \rightsquigarrow g' \;\;|\;\; a \Vdash_{g'} N(\Gamma; \sigma)}{a \Vdash_g N(\Gamma; \sigma)}\,\text{N-N}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{App}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau, \diamond \Rightarrow \sigma) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a; \mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b; \mathcal{M})}{ab \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\text{APP}$$

$$\frac{n \in [\![0, \mathrm{sz}(\rho)]\!],\; \mathcal{M} \Vvdash S(\rho, [\diamond_1, \ldots, \diamond_n]) \quad | \quad a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y : \mathcal{M}; \sigma) \quad b \Vdash_{g_b} H^{z:[\rho]}(\Gamma_b; \mathcal{M})}{a[y \backslash b] \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\text{ES-H}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \mathtt{fix}\;\; y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in [\![1, \mathrm{sz}(\tau)]\!],\; [\rho_i]_{i \in [\![1,n]\!]} \Vvdash S(\tau, [\diamond_1, \ldots, \diamond_n]) \\ j \in [\![1, n]\!],\; \sigma \Vvdash S(\rho_j, \diamond) \end{array} \;\Bigg|\; a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y : [\rho_i]_{i \in [\![1,n]\!] \backslash j}; \sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b; [\rho_i]_{i \in [\![1,n]\!]})}{a[y \backslash b] \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\text{ES-CH}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\tau], \qquad \mathtt{fix}\;\; y \notin \mathrm{dom}(\Gamma) \\ n \in [\![0, \mathrm{sz}(\tau)]\!],\; \mathcal{M} \Vvdash S(\tau, [\diamond_1, \ldots, \diamond_n]) \end{array} \;\Bigg|\; a \Vdash_{g_a} N(\Gamma_a, y : \mathcal{M}; \sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b; \mathcal{M})}{a[y \backslash b] \Vdash_g N(\Gamma; \sigma)}\,\text{ES-N}$$

The Full Algorithm

$$\frac{g \rightsquigarrow \mathsf{Var} \quad \mid}{x \Vdash_g H^{x:[\sigma]}(\emptyset; \sigma)}\,\textsf{VAR} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Der}(g') \\ [\sigma] \Vvdash S(\tau, \diamond) \end{array} \;\middle|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma; [\sigma])}{\mathsf{der}(a) \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\textsf{DR}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{App}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b \\ \mathcal{M} \Rightarrow \sigma \Vvdash S(\tau, \diamond \Rightarrow \sigma) \end{array} \;\middle|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a; \mathcal{M} \Rightarrow \sigma) \quad b \Vdash_{g_b} N(\Gamma_b; \mathcal{M})}{ab \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\textsf{APP}$$

$$\frac{g \rightsquigarrow g' \;\mid\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma; \sigma)}{a \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\textsf{H-H} \qquad \frac{\begin{array}{c} g \rightsquigarrow g' \\ \Gamma = \Gamma' + x : [\tau] \\ \sigma \Vvdash S(\tau, \diamond) \end{array} \;\middle|\; a \Vdash_{g'} H^{x:[\tau]}(\Gamma'; \sigma)}{a \Vdash_g N(\Gamma; \sigma)}\,\textsf{N-H} \qquad \frac{g \rightsquigarrow g' \;\mid\; a \Vdash_{g'} N(\Gamma; \sigma)}{a \Vdash_g N(\Gamma; \sigma)}\,\textsf{N-N}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Lam}(g') \\ \mathtt{fix}\ \ x \notin \mathrm{dom}(\Gamma) \end{array} \;\middle|\; a \Vdash_{g'} N(\Gamma, x : \mathcal{M}; \sigma)}{\lambda x.a \Vdash_g N(\Gamma; \mathcal{M} \Rightarrow \sigma)}\,\textsf{ABS} \qquad \frac{\begin{array}{c} g \rightsquigarrow \mathsf{Bng}(g') \\ I \neq \emptyset \\ \Gamma = +_{i \in I} \Gamma_i \end{array} \;\middle|\; \left(a_i \Vdash_{g'} N(\Gamma_i; \tau_i)\right)_{i \in I} \quad \uparrow_{i \in I} a_i}{! \bigvee_{i \in I} a_i \Vdash_g N(\Gamma; [\tau_i]_{i \in I})}\,\textsf{BG} \qquad \frac{g \rightsquigarrow \mathsf{Bng}(\bot) \quad \mid}{!\bot \Vdash_g N(\emptyset; [\,])}\,\textsf{BG}_{\bot}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\rho], \qquad \mathtt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in \llbracket 0, \mathrm{sz}(\rho) \rrbracket,\ \mathcal{M} \Vvdash S(\rho, [\diamond_1, \ldots, \diamond_n]) \end{array} \;\middle|\; a \Vdash_{g_a} H^{x:[\tau]}(\Gamma_a, y{:}\,\mathcal{M}; \sigma) \quad b \Vdash_{g_b} H^{z:[\rho]}(\Gamma_b; \mathcal{M})}{a[y \backslash b] \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\textsf{ES-H}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b, \qquad \mathtt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in \llbracket 1, \mathrm{sz}(\tau) \rrbracket,\ [\rho_i]_{i \in \llbracket 1, n \rrbracket} \Vvdash S(\tau, [\diamond_1, \ldots, \diamond_n]) \\ j \in \llbracket 1, n \rrbracket,\ \sigma \Vvdash S(\rho_j, \diamond) \end{array} \;\middle|\; a \Vdash_{g_a} H^{y:[\rho_j]}(\Gamma_a, y : [\rho_i]_{i \in \llbracket 1, n \rrbracket \backslash j}; \sigma) \quad b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b; [\rho_i]_{i \in \llbracket 1, n \rrbracket})}{a[y \backslash b] \Vdash_g H^{x:[\tau]}(\Gamma; \sigma)}\,\textsf{ES-CH}$$

$$\frac{\begin{array}{c} g \rightsquigarrow \mathsf{Sub}(g_a, g_b) \\ \Gamma = \Gamma_a + \Gamma_b + z : [\tau], \qquad \mathtt{fix}\ \ y \notin \mathrm{dom}(\Gamma) \\ n \in \llbracket 0, \mathrm{sz}(\tau) \rrbracket,\ \mathcal{M} \Vvdash S(\tau, [\diamond_1, \ldots, \diamond_n]) \end{array} \;\middle|\; a \Vdash_{g_a} N(\Gamma_a, y : \mathcal{M}; \sigma) \quad b \Vdash_{g_b} H^{z:[\tau]}(\Gamma_b; \mathcal{M})}{a[y \backslash b] \Vdash_g N(\Gamma; \sigma)}\,\textsf{ES-N}$$

The Full Algorithm and its Implementation

$$\frac{g \rightsquigarrow \mathsf{Var}}{x \Vdash_g H^{x:[\sigma]}(\emptyset;\sigma)}\ \text{VAR}$$

$$g \rightsquigarrow \mathsf{Der}(g') \qquad [\sigma] \Vdash S(\tau,\diamond) \quad \Big| \quad a \Vdash_{g'} H^{x:[\tau]}(\Gamma;[$$

$$\mathsf{der}(a) \Vdash$$

$$g \rightsquigarrow \mathsf{App}(g_a, g_b) \qquad \Gamma = \Gamma_a + \Gamma_b \qquad \mathcal{M} \Rightarrow \sigma \Vdash S($$

g fix

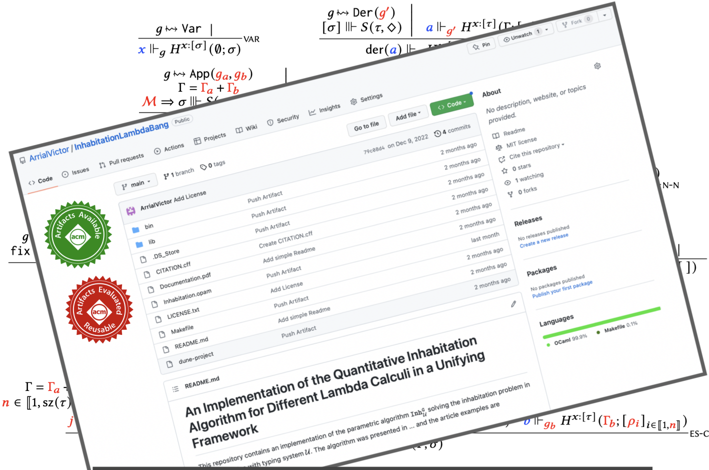
N-N

BG$_\perp$ $[\,])$

$\Gamma = \Gamma_a$ $n \in [\![1, \mathrm{sz}(\tau)$ j

$b \Vdash_{g_b} H^{x:[\tau]}(\Gamma_b; [\rho_i]_{i\in[\![1,n]\!]})$ ES-CH

(τ, σ)



ArrialVictor / InhabitationLambdaBang Public

Pin · Unwatch 1 · Fork 0

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

main · 1 branch · 0 tags · Go to file · Add file · Code

ArrialVictor Add License — 79c08d4 on Dec 9, 2022 — 4 commits

bin	Push Artifact	2 months ago
lib	Push Artifact	2 months ago
.DS_Store	Push Artifact	2 months ago
CITATION.cff	Create CITATION.cff	2 months ago
Documentation.pdf	Add simple Readme	2 months ago
Inhabitation.opam	Push Artifact	2 months ago
LICENSE.txt	Add License	last month
Makefile	Push Artifact	2 months ago
README.md	Add simple Readme	2 months ago
dune-project	Push Artifact	2 months ago

About

No description, website, or topics provided.

Readme · MIT license · Cite this repository · 0 stars · 1 watching · 0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

OCaml 99.9% · Makefile 0.1%

README.md

An Implementation of the Quantitative Inhabitation Algorithm for Different Lambda Calculi in a Unifying Framework

This repository contains an implementation of the parametric algorithm $\mathtt{Inh}^{\mathcal{G}}_{\mathcal{U}}$ solving the inhabitation problem in ... with typing system $\mathcal{U}$. The algorithm was presented in ... and the article examples are

github/ArrialVictor/InhabitationLambdaBang

Properties of the Inhabitation Algorithm

Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Theorem

✓ *The inhabitation algorithm terminates.*

Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete (i.e. it exactly computes* $\mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$*).*

Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete (i.e. it exactly computes* $\mathtt{Basis}_{\mathcal{B}}(\Gamma,\sigma)$*).*

More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the $\boxed{\textbf{BANG}}$ IP.

Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete (i.e. it exactly computes $\mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$).*

More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the BANG IP.
- Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Solving NAME Inhabitation - Standard Methology

Solving NAME Inhabitation - Standard Methology

Theorem ([BucciarelliKesnerRios14])

✓ *For any typing* (Γ, σ), $\texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma)$ ***exists***, *is* ***finite***, ***correct*** *and* ***complete***. NAME

Solving NAME Inhabitation - Standard Methology

Theorem ([BucciarelliKesnerRios14])

✓ *For any typing* (Γ, σ), $\mathtt{Basis}_{\mathcal{N}}(\Gamma, \sigma)$ ***exists**, is **finite**, **correct** and **complete**.* NAME

Built an algorithm computing $\mathtt{Basis}_{\mathcal{N}}(\Gamma, \sigma)$: **[BucciarelliKesnerRios14]**

$$\frac{\mathsf{a} \Vdash \mathtt{T}(\Gamma + \mathtt{x} : \mathtt{A}, \tau) \qquad \mathtt{x} \notin \mathtt{dom}(\Gamma)}{\lambda \mathtt{x}.\mathsf{a} \Vdash \mathtt{T}(\Gamma, \mathtt{A} \to \tau)}\ (\mathtt{Abs})$$

$$\frac{(\mathsf{a}_i \Vdash \mathtt{T}(\Gamma_i, \sigma_i))_{i \in I} \qquad \uparrow_{i \in I} \mathsf{a}_i}{\bigvee_{i \in I} \mathsf{a}_i \Vdash \mathtt{TI}(+_{i \in I} \Gamma_i, [\sigma_i]_{i \in I})}\ (\mathtt{Union})$$

$$\frac{\Gamma = \Gamma_1 + \Gamma_2 \quad \mathsf{a} \Vdash \mathtt{H}^{\mathtt{x}:[\mathtt{A}_1 \to \ldots \mathtt{A}_n \to \mathtt{B} \to \tau]}(\Gamma_1, \mathtt{B} \to \tau) \quad \mathsf{b} \Vdash \mathtt{TI}(\Gamma_2, \mathtt{B}) \quad n \geq 0}{\mathsf{ab} \Vdash \mathtt{H}^{\mathtt{x}:[\mathtt{A}_1 \to \ldots \mathtt{A}_n \to \mathtt{B} \to \tau]}(\Gamma, \tau)}\ (\mathtt{Head}_{>0})$$

$$\frac{}{\mathtt{x} \Vdash \mathtt{H}^{\mathtt{x}:[\tau]}(\emptyset, \tau)}\ (\mathtt{Head}_0)$$

$$\frac{\mathsf{a} \Vdash \mathtt{H}^{\mathtt{x}:[\mathtt{A}_1 \to \ldots \mathtt{A}_n \to \tau]}(\Gamma, \tau)}{\mathsf{a} \Vdash \mathtt{T}(\Gamma + \mathtt{x} : [\mathtt{A}_1 \to \ldots \mathtt{A}_n \to \tau], \tau)}\ (\mathtt{Head})$$

Solving NAME Inhabitation : through BANG Inhabitation

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $t \in \mathtt{Basis}_{\mathcal{N}}(\Gamma, \sigma)$

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

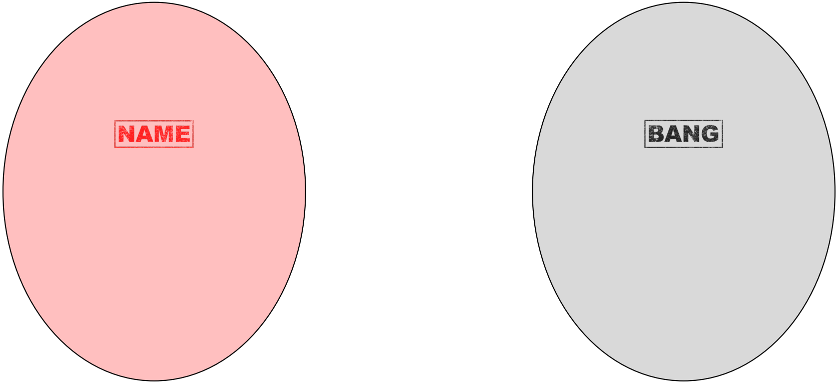
NAME $t \in \mathtt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $\quad t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

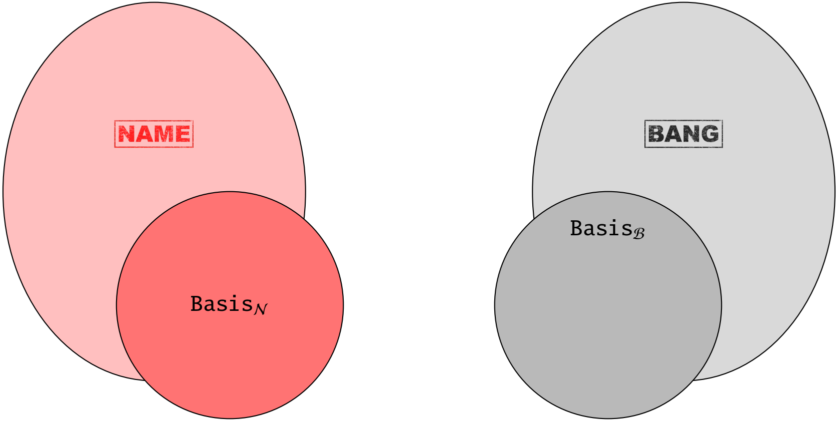


Solving NAME Inhabitation : through BANG Inhabitation

The Basis is preserved by the embedding:

Theorem

NAME $\quad t \in \mathtt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

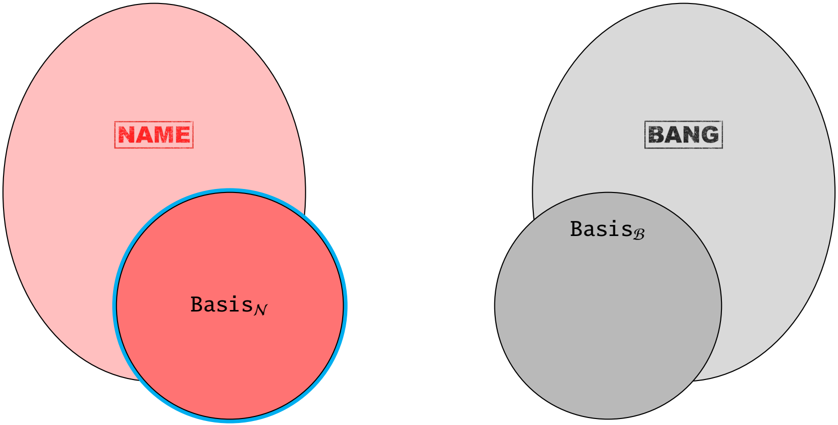

NAME
$\mathtt{Basis}_{\mathcal{N}}$
BANG
$\mathtt{Basis}_{\mathcal{B}}$

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $\quad t \in \mathtt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \quad \Leftrightarrow \quad t^{N} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

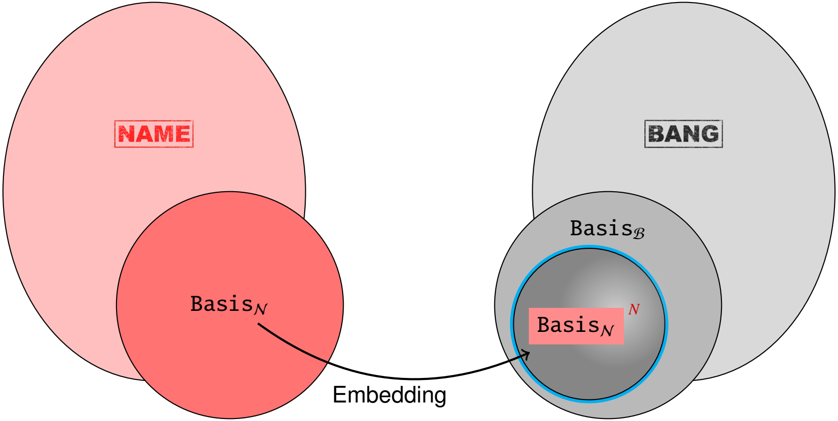

NAME
$\mathtt{Basis}_{\mathcal{N}}$
BANG
$\mathtt{Basis}_{\mathcal{B}}$

Solving NAME Inhabitation : through BANG Inhabitation

The Basis is preserved by the embedding:

Theorem

NAME $t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

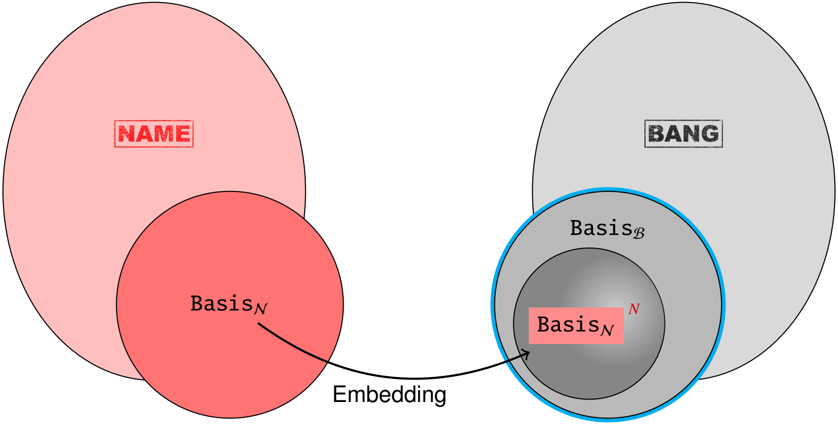

NAME
$\texttt{Basis}_{\mathcal{N}}$
BANG
$\texttt{Basis}_{\mathcal{B}}$
$\texttt{Basis}_{\mathcal{N}}^{N}$
Embedding

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $\quad t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

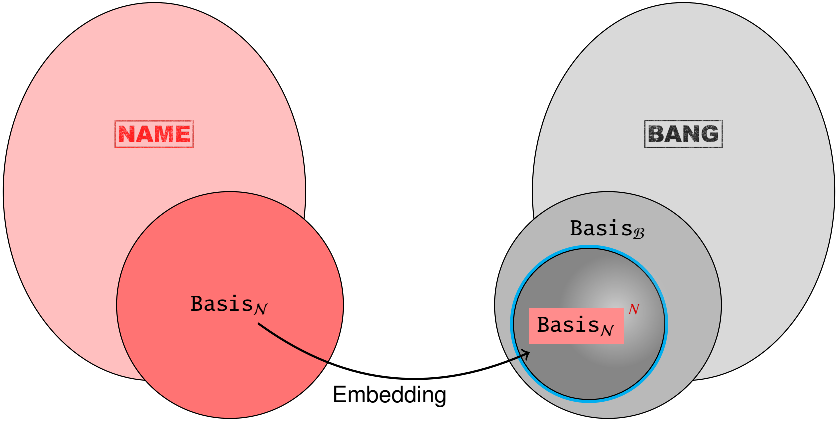

NAME
BANG
$\texttt{Basis}_{\mathcal{B}}$
$\texttt{Basis}_{\mathcal{N}}$
$\texttt{Basis}_{\mathcal{N}}^{N}$
Embedding

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $\quad t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \quad \Leftrightarrow \quad t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

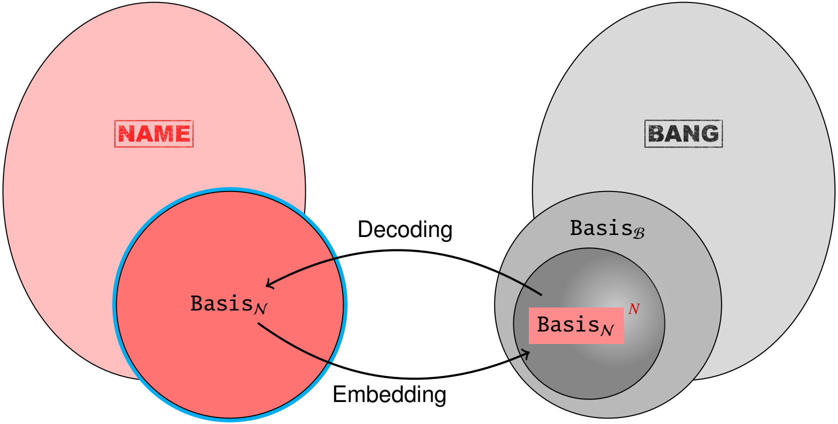

NAME
BANG
$\texttt{Basis}_{\mathcal{N}}$
$\texttt{Basis}_{\mathcal{B}}$
$\texttt{Basis}_{\mathcal{N}}^{N}$
Embedding

Solving NAME Inhabitation : through BANG Inhabitation

The $\texttt{Basis}$ is preserved by the embedding:

Theorem

NAME $\quad t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

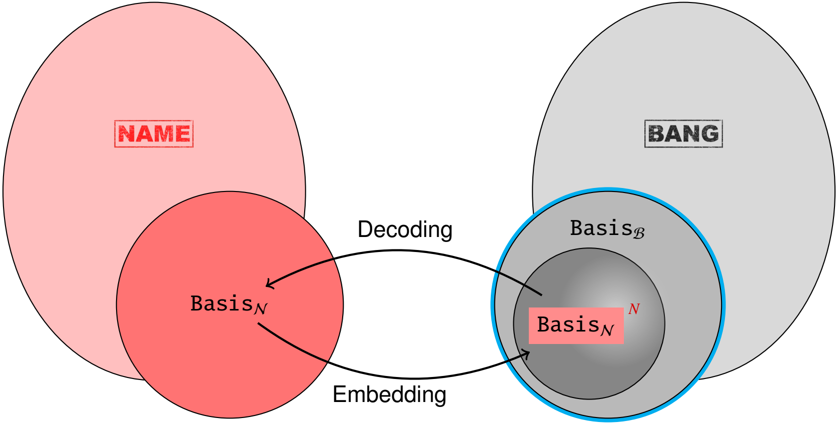


Solving NAME Inhabitation : through BANG Inhabitation

The Basis is preserved by the embedding:

Theorem

NAME $t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

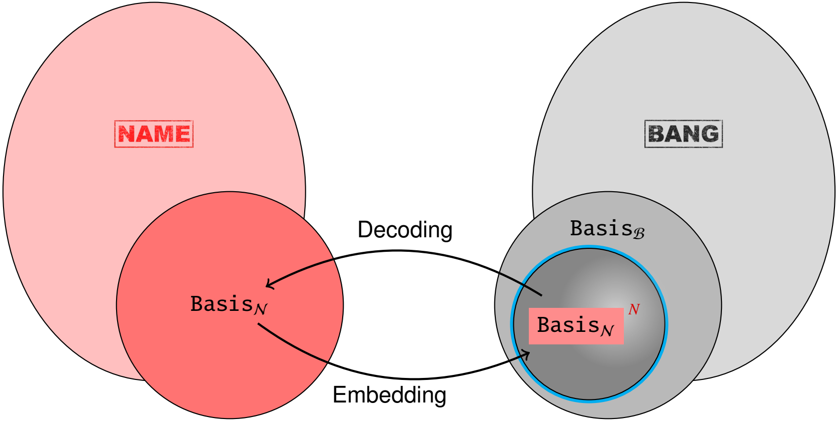


Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $$t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{N} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$$ BANG

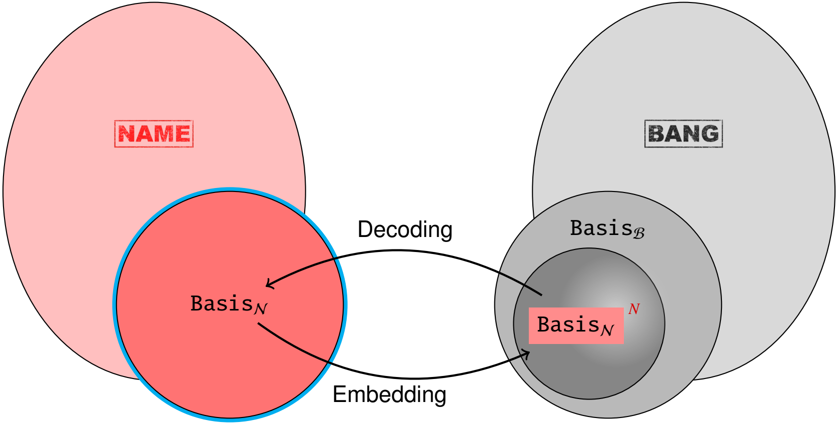

NAME
BANG
$\texttt{Basis}_{\mathcal{B}}$
Decoding
$\texttt{Basis}_{\mathcal{N}}$
$\texttt{Basis}_{\mathcal{N}}^{N}$
Embedding

Solving NAME Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

NAME $t \in \texttt{Basis}_{\mathcal{N}}(\Gamma, \sigma) \Leftrightarrow t^{\mathcal{N}} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG


NAME
BANG
$\texttt{Basis}_{\mathcal{B}}$
Decoding
$\texttt{Basis}_{\mathcal{N}}$
$\texttt{Basis}_{\mathcal{N}}^{\mathcal{N}}$
Embedding

Solving VALUE Inhabitation - Usual Methology

Solving VALUE Inhabitation - Usual Methology

Theorem

✓ *For any typing* (Γ, σ), $\texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma)$ ***exists***, *is* ***finite***, ***correct*** *and* ***complete***. VALUE

Solving VALUE Inhabitation - Usual Methology

Theorem

✓ *For any typing* (Γ, σ), $\mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma)$ ***exists****, is* ***finite****,* ***correct*** *and* ***complete****.* VALUE

Built an algorithm computing $\mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma)$:

Solving VALUE Inhabitation - Usual Methology

Theorem

- *For any typing* (Γ,σ), $\mathtt{Basis}_{\mathcal{V}}(\Gamma,\sigma)$ ***exists**, is **finite**, **correct** and **complete**.* VALUE

Built an algorithm computing $\mathtt{Basis}_{\mathcal{V}}(\Gamma,\sigma)$:

$$\dfrac{\;\mid\;}{x \Vdash H_{\mathrm{F}}^{x:[\sigma]}(\emptyset;\sigma)}\,\text{VAR-FUN} \qquad \dfrac{\begin{array}{c} I \neq \emptyset \\ \Gamma = x : [\sigma_i]_{i\in I}\end{array} \;\Big|\;}{x \Vdash N(\Gamma;[\sigma]_{i\in I})}\,\text{VAR-VAL} \qquad \dfrac{\;\mid\;}{\bot_{\mathsf{v}} \Vdash N(\emptyset;[\,])}\,\text{VAR}_{\bot}$$

$$\dfrac{\begin{array}{c} \Gamma = \Gamma_1 + \Gamma_2 \\ [\mathcal{M} \Rightarrow \sigma] \Vvdash S(\tau,[\diamond \Rightarrow \sigma])\end{array} \;\Big|\; a_1 \Vdash H_{Q}^{x:[\tau]}(\Gamma_1;[\mathcal{M} \Rightarrow \sigma]) \quad a_2 \Vdash N(\Gamma_2;\mathcal{M})}{a_1 a_2 \Vdash H_{A}^{x:[\tau]}(\Gamma;\sigma)}\,\text{APP}_{Q} \qquad \dfrac{\;\mid\;}{\lambda x.\bot \Vdash N(\emptyset;[\,])}\,\text{ABS}_{\bot}$$

$$\dfrac{\begin{array}{c} \Gamma = \Gamma' + x : [\tau] \\ \sigma \Vvdash S(\tau,\diamond)\end{array} \;\Big|\; a \Vdash H_{A}^{x:[\tau]}(\Gamma';\sigma)}{a \Vdash N(\Gamma;\sigma)}\,\text{N-H}_{A} \qquad \dfrac{\begin{array}{c} I \neq \emptyset \\ \Gamma = +_{i\in I}\Gamma_i \\ \mathtt{fix}\ \ x \in \mathrm{dom}(\Gamma)\end{array} \;\Big|\; (\, a_i \Vdash N(\Gamma_i, x : \mathcal{M}_i;\sigma_i))_{i\in I} \quad \uparrow_{i\in I} a_i}{\lambda x. \bigvee_{i\in I} a_i \Vdash N(\Gamma;[\mathcal{M}_i \Rightarrow \sigma_i]_{i\in i})}\,\text{ABS}$$

$$\dfrac{\begin{array}{c} \Gamma = \Gamma_a + \Gamma_b + z : [\rho], \quad \mathtt{fix}\ y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in [\![0,\mathrm{sz}(\rho)]\!],\ \mathcal{M} \Vvdash S(\rho,[\diamond_1,\ldots,\diamond_n])\end{array} \;\Big|\; a \Vdash H_{Q}^{x:[\tau]}(\Gamma_a, y : \mathcal{M};\sigma) \quad b \Vdash H_{A}^{z:[\rho]}(\Gamma_b;\mathcal{M})}{a[y\backslash b] \Vdash H_{Q}^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-H}_{Q}$$

$$\dfrac{\begin{array}{c} \Gamma = \Gamma_a + \Gamma_b, \quad \mathtt{fix}\ y \notin \mathrm{dom}(\Gamma) \cup \{x\} \\ n \in [\![1,\mathrm{sz}(\tau)]\!],\ [\rho_i]_{i\in[\![1,n]\!]} \Vvdash S(\tau,[\diamond_1,\ldots,\diamond_n]) \\ j \in [\![1,n]\!],\ \sigma \Vvdash S(\rho_j,\diamond)\end{array} \;\Bigg|\; a \Vdash H_{Q}^{y:[\rho_j]}(\Gamma_a, y : [\rho_i]_{i\in[\![1,n]\!]\backslash j};\sigma) \quad b \Vdash H_{A}^{x:[\tau]}(\Gamma_b;[\rho_i]_{i\in[\![1,n]\!]})}{a[y\backslash b] \Vdash H_{Q}^{x:[\tau]}(\Gamma;\sigma)}\,\text{ES-CH}_{Q}$$

$$\dfrac{\begin{array}{c} \Gamma = \Gamma_a + \Gamma_b + z : [\tau], \quad \mathtt{fix}\ y \notin \mathrm{dom}(\Gamma) \\ n \in [\![0,\mathrm{sz}(\tau)]\!],\ \mathcal{M} \Vvdash S(\tau,[\diamond_1,\ldots,\diamond_n])\end{array} \;\Big|\; a \Vdash N(\Gamma_a, y : \mathcal{M};\sigma) \quad b \Vdash H_{A}^{z:[\tau]}(\Gamma_b;\mathcal{M})}{a[y\backslash b] \Vdash N(\Gamma;\sigma)}\,\text{ES-N}$$

Solving VALUE Inhabitation : through BANG Inhabitation

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma)$

Solving VALUE Inhabitation : through BANG Inhabitation

The Basis is preserved by the embedding:

Theorem

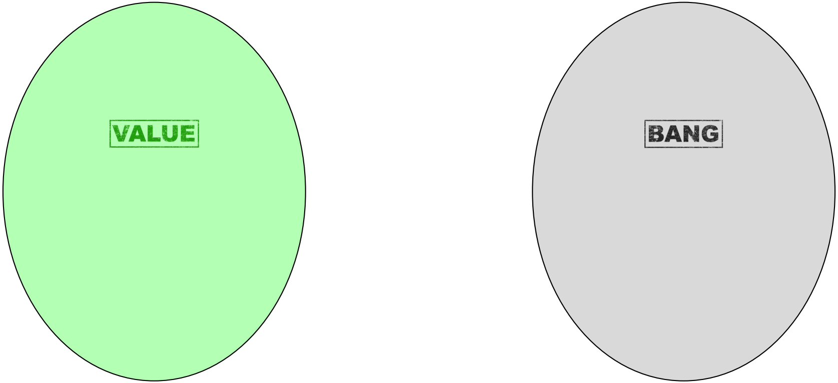
VALUE $t \in \mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $\quad t \in \texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \quad \Leftrightarrow \quad t^{V} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

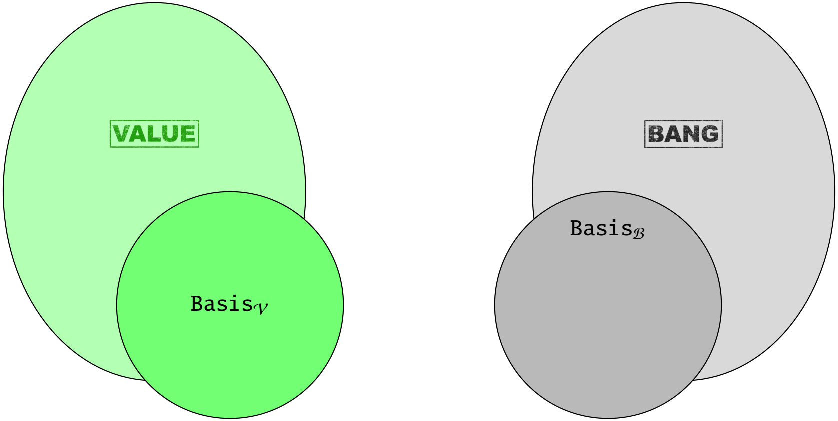


Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

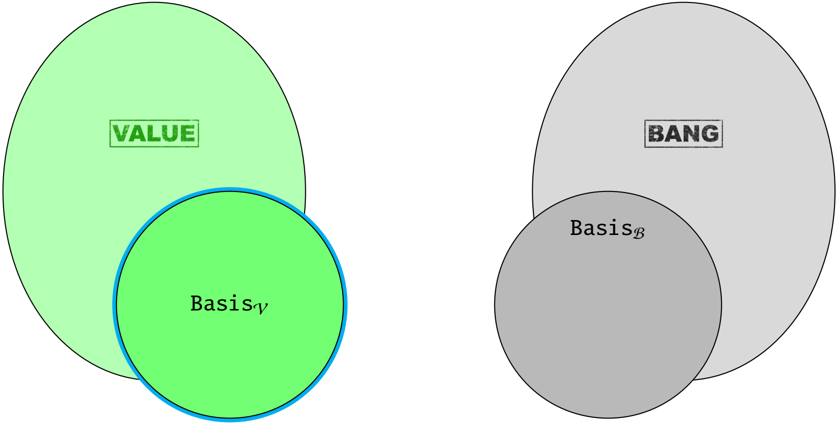

VALUE
$\mathtt{Basis}_{\mathcal{V}}$
BANG
$\mathtt{Basis}_{\mathcal{B}}$

Solving VALUE Inhabitation : through BANG Inhabitation

The Basis is preserved by the embedding:

Theorem

VALUE $t \in \texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

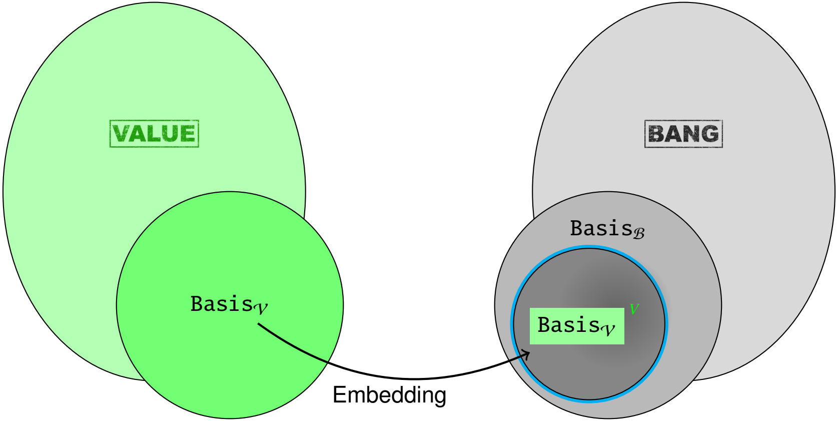

VALUE
$\texttt{Basis}_{\mathcal{V}}$
BANG
$\texttt{Basis}_{\mathcal{B}}$

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

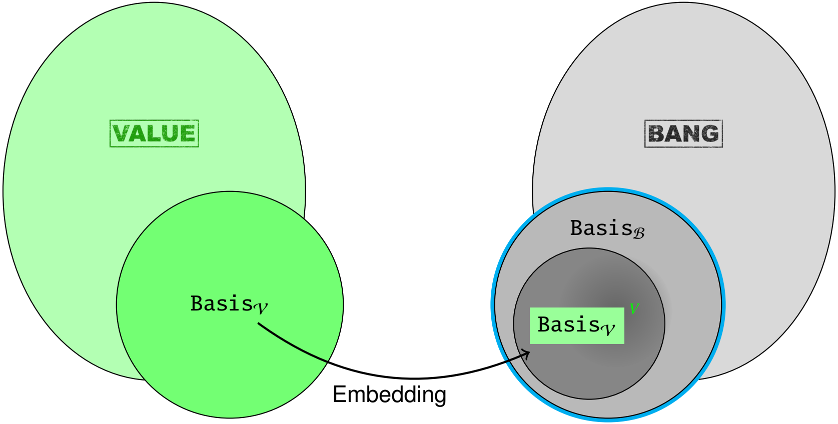

VALUE
$\mathtt{Basis}_{\mathcal{V}}$
BANG
$\mathtt{Basis}_{\mathcal{B}}$
$\mathtt{Basis}_{\mathcal{V}}^{V}$
Embedding

Solving VALUE Inhabitation : through BANG Inhabitation

The Basis is preserved by the embedding:

Theorem

VALUE $\quad t \in \texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma) \quad$ BANG

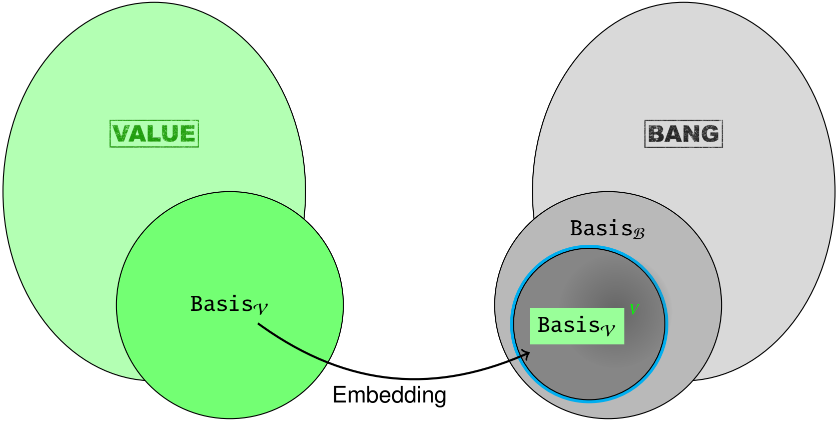

VALUE
$\texttt{Basis}_{\mathcal{V}}$
BANG
$\texttt{Basis}_{\mathcal{B}}$
$\texttt{Basis}_{\mathcal{V}}^{V}$
Embedding

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma)$ $\Leftrightarrow$ $t^{V} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

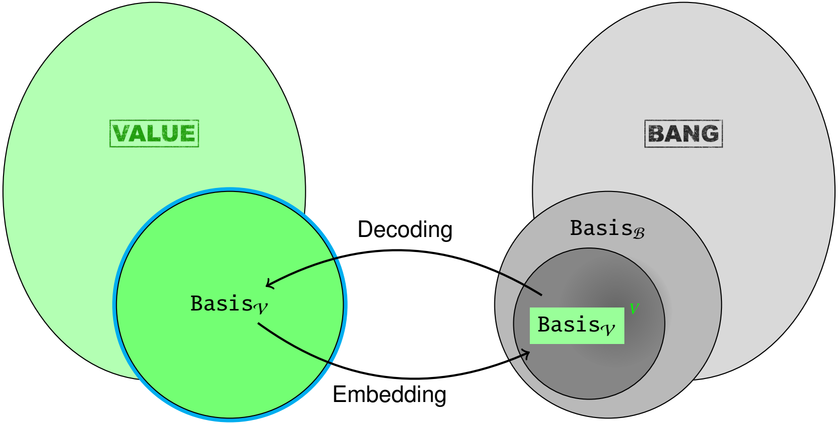

VALUE
$\mathtt{Basis}_{\mathcal{V}}$
Embedding
BANG
$\mathtt{Basis}_{\mathcal{B}}$
$\mathtt{Basis}_{\mathcal{V}}^{V}$

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

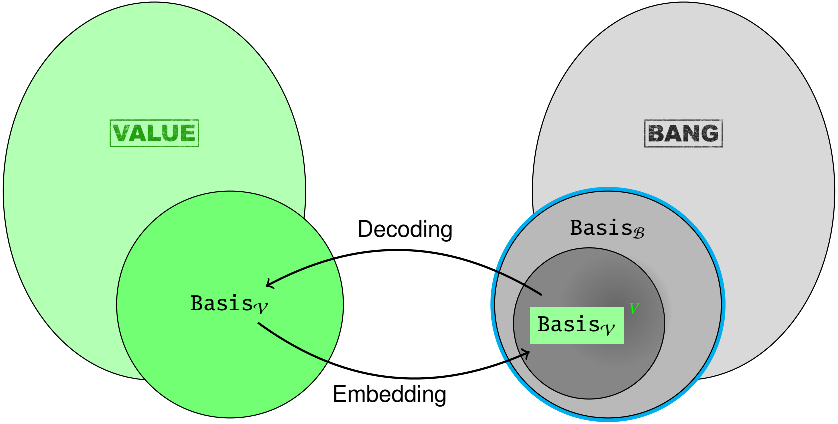

VALUE
BANG
$\texttt{Basis}_{\mathcal{B}}$
Decoding
$\texttt{Basis}_{\mathcal{V}}$
$\texttt{Basis}_{\mathcal{V}}^{V}$
Embedding

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \mathtt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG

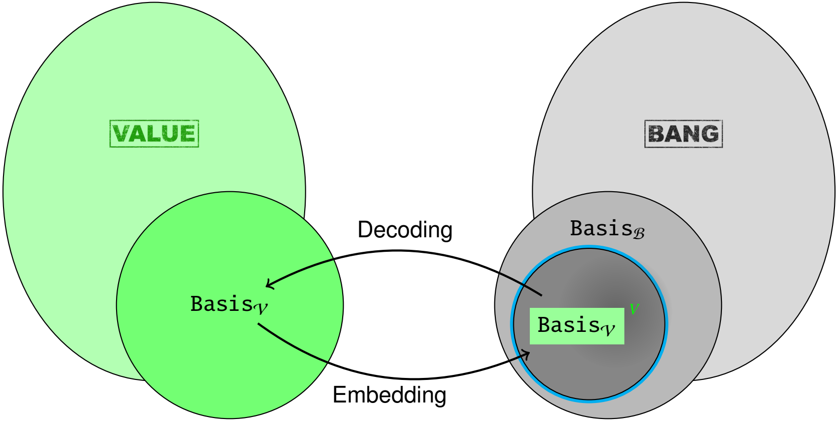

VALUE
BANG
$\mathtt{Basis}_{\mathcal{B}}$
Decoding
$\mathtt{Basis}_{\mathcal{V}}$
$\mathtt{Basis}_{\mathcal{V}}^{V}$
Embedding

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

$$\text{VALUE} \qquad t \in \texttt{Basis}_{\mathcal{V}}(\Gamma,\sigma) \quad \Leftrightarrow \quad t^{V} \in \texttt{Basis}_{\mathcal{B}}(\Gamma,\sigma) \qquad \text{BANG}$$

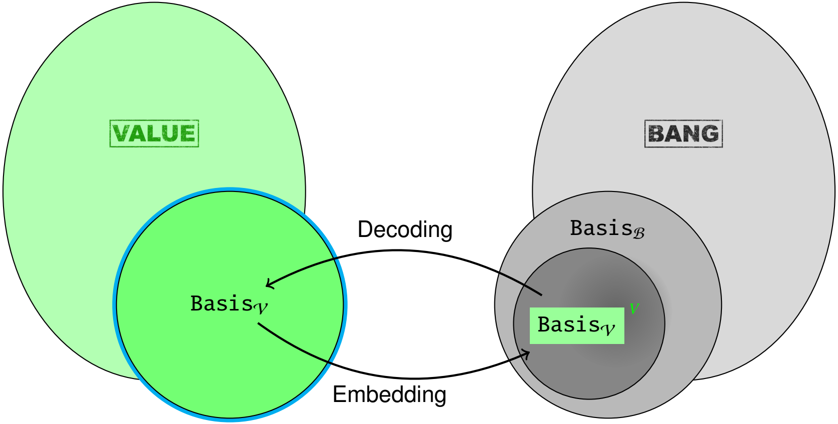

VALUE
BANG
$\texttt{Basis}_{\mathcal{B}}$
Decoding
$\texttt{Basis}_{\mathcal{V}}$
$\texttt{Basis}_{\mathcal{V}}{}^{V}$
Embedding

Solving VALUE Inhabitation : through BANG Inhabitation

The `Basis` is preserved by the embedding:

Theorem

VALUE $t \in \texttt{Basis}_{\mathcal{V}}(\Gamma, \sigma) \Leftrightarrow t^{V} \in \texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$ BANG



Properties of the Indirect NAME and VALUE Algorithm

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete*
  (i.e. it exactly computes $\texttt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$*).*

BANG

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete*
  (i.e. it exactly computes $\mathtt{Basis}_{\mathcal{B}}(\Gamma, \sigma)$*).*

BANG

More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the BANG IP.
- Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete*
  (i.e. it exactly computes `Basis` (Γ, σ)*).*

BANG
⟶
NAME
VALUE

More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the BANG IP.
- Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete*
  (i.e. it exactly computes $\texttt{Basis}\ (\Gamma, \sigma)$*).*

BANG ⟶ NAME VALUE

More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the BANG IP.
- ✓ Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete (i.e. it exactly computes* $\mathtt{Basis}\ (\Gamma, \sigma)$*).*

BANG ⟶ NAME VALUE OTHERS

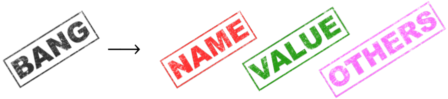
More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the BANG IP.
- ✓ Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- *The inhabitation algorithm terminates.*
- *The algorithm is sound and complete*
  (i.e. it exactly computes $\texttt{Basis}\ (\Gamma, \sigma)$*).*



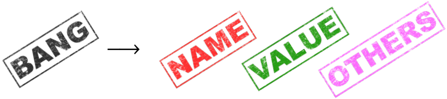
More Ambitious Third Goal

- Decidability by **finding all inhabitants** in the BANG IP.
- Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- Using generic properties so that other encodable models of computation can use these results.

Properties of the Indirect NAME and VALUE Algorithm

Theorem

- ✓ *The inhabitation algorithm terminates.*
- ✓ *The algorithm is sound and complete (i.e. it exactly computes* $\texttt{Basis}\ (\Gamma, \sigma)$*).*



More Ambitious Third Goal

- ✓ Decidability by **finding all inhabitants** in the BANG IP.
- ✓ Decidability of the NAME and VALUE IP by **finding all inhabitants** from those of the BANG IP.
- ✓ Using generic properties so that other encodable models of computation can use these results.

Artifacts Available acm

Artifacts Evaluated Reusable acm

Conclusion

Conclusion

Summary:

- Solving the generalized inhabitation problem
- A several-for-one deal: BANG NAME VALUE OTHERS
- An implementation: (`github/ArrialVictor/InhabitationLambdaBang`)

Conclusion

Summary:

- Solving the generalized inhabitation problem
- A several-for-one deal: BANG NAME VALUE OTHERS
- An implementation: (`github/ArrialVictor/InhabitationLambdaBang`)

Further questions and ongoing work:

- Solvability (for Different Calculi in a Unified Framework)

Conclusion

Summary:

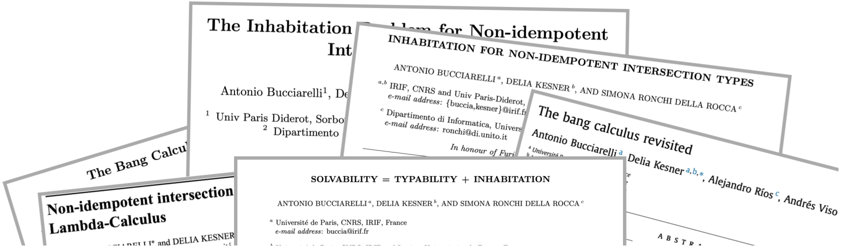
- Solving the generalized inhabitation problem
- A several-for-one deal: BANG NAME VALUE OTHERS
- An implementation: (`github/ArrialVictor/InhabitationLambdaBang`)

Further questions and ongoing work:

- Solvability (for Different Calculi in a Unified Framework)

Thanks for your attention!

Thank you !



The Inhabitation Problem for Non-idempotent
Int

Antonio Bucciarelli[1], De

[1] Univ Paris Diderot, Sorbo
[2] Dipartimento

INHABITATION FOR NON-IDEMPOTENT INTERSECTION TYPES

ANTONIO BUCCIARELLI [a], DELIA KESNER [b], AND SIMONA RONCHI DELLA ROCCA [c]

[a,b] IRIF, CNRS and Univ Paris-Diderot,
e-mail address: {buccia,kesner}@irif.fr

[c] Dipartimento di Informatica, Univers
e-mail address: ronchi@di.unito.it

In honour of Furi

The bang calculus revisited

Antonio Bucciarelli [a], Delia Kesner [a,b,*], Alejandro Ríos [c], Andrés Viso

ABSTRA

The Bang Calcul

Non-idempotent intersection
Lambda-Calculus

and DELIA KESNER

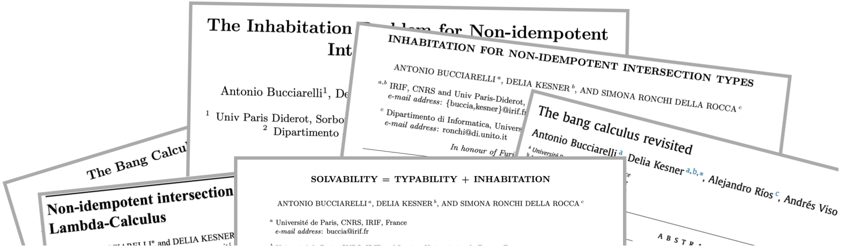
SOLVABILITY = TYPABILITY + INHABITATION

ANTONIO BUCCIARELLI [a], DELIA KESNER [b], AND SIMONA RONCHI DELLA ROCCA [c]

[a] Université de Paris, CNRS, IRIF, France
e-mail address: buccia@irif.fr

Happy Birthday !

Thank you !



The Inhabitation Problem for Non-idempotent Int

Antonio Bucciarelli[1], De

[1] Univ Paris Diderot, Sorbo

[2] Dipartimento

INHABITATION FOR NON-IDEMPOTENT INTERSECTION TYPES

ANTONIO BUCCIARELLI[a], DELIA KESNER[b], AND SIMONA RONCHI DELLA ROCCA[c]

[a,b] IRIF, CNRS and Univ Paris-Diderot,
e-mail address: {buccia,kesner}@irif.fr

[c] Dipartimento di Informatica, Univers
e-mail address: ronchi@di.unito.it

In honour of Furi

The bang calculus revisited

Antonio Bucciarelli[a], Delia Kesner[a,b,*], Alejandro Ríos[c], Andrés Viso

[a] Université

ABSTRA

The Bang Calcul

Non-idempotent intersection Lambda-Calculus

and DELIA KESNER

SOLVABILITY = TYPABILITY + INHABITATION

ANTONIO BUCCIARELLI[a], DELIA KESNER[b], AND SIMONA RONCHI DELLA ROCCA[c]

[a] Université de Paris, CNRS, IRIF, France
e-mail address: buccia@irif.fr