Bang Calculus: A Subsuming Paradigm

**Static Properties:**
[BucciarelliKesnerRíosViso'20,'23]

**Dynamic Properties:**
[BucciarelliKesnerRíosViso'20,'23]
Bang Calculus: A Subsuming Paradigm

Static Properties:
[BucciarelliKesnerRíosViso’20,’23]

Dynamic Properties:
[BucciarelliKesnerRíosViso’20,’23]
Bang Calculus: A Subsuming Paradigm

Static Properties:

Dynamic Properties:

\[\text{BucciarelliKesnerR´ıosViso'20,'23}\]
Bang Calculus: A Subsuming Paradigm

Static Properties:

$NF_{CBN}$

Dynamic Properties:

$\downarrow u, \downarrow N_{CBN}$

$\downarrow u, \downarrow V$
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

\[
\text{NAME} \quad t \quad N \quad \text{normal form} \quad \Rightarrow \quad t^N \quad \text{normal form} \quad \text{BANG}
\]
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

$$t \text{ normal form} \iff t^N \text{ normal form}$$
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ \text{NAME} \quad t \text{ normal form} \iff t^N \text{ normal form} \]

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME**
- $t$ normal form $\iff t^N$ normal form

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME**
- $t \rightarrow u$
Bang Calculus: A Subsuming Paradigm

**Static Properties:** [BucciarelliKesnerRíosViso’20,’23]

\[ t \text{ normal form} \iff t^N \text{ normal form} \]

**Dynamic Properties:** [BucciarelliKesnerRíosViso’20,’23]

\[ t \rightarrow u \]
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ t \text{ normal form} \iff t^N \text{ normal form} \]

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ t \rightarrow u \implies t^N \rightarrow u^N \]
Static Properties: [BucciarelliKesnerRiosViso'20,'23]

\[ \text{Name} \quad t \text{ normal form} \iff t^N \text{ normal form} \]

Dynamic Properties: [BucciarelliKesnerRiosViso'20,'23]

\[ \text{Name} \quad t \rightarrow u \quad \Rightarrow \quad t^N \rightarrow u^N \]

Bang Calculus: A Subsuming Paradigm
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ t \text{ normal form} \Leftrightarrow t^N \text{ normal form} \]

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ t \rightarrow u \quad \Rightarrow \quad t^N \rightarrow u^N \]
Static Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ t \text{ normal form} \iff t^N \text{ normal form} \]

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

\[ t \to u \implies t^N \to u^N \]
Bang Calculus: A Subsuming Paradigm

**Static Properties:** [BucciarelliKesnerRíosViso’20,’23]

- **NAME**
  - $t$ normal form $\Leftrightarrow t^N$ normal form
- **VALUE**
  - $t$ normal form $\Rightarrow t^V$ normal form

**Dynamic Properties:** [BucciarelliKesnerRíosViso’20,’23]

- **NAME**
  - $t \rightarrow u$ $\Rightarrow t^N \rightarrow u^N$
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME** \( t \) normal form \( \Leftrightarrow \) \( t^N \) normal form
- **VALUE** \( t \) normal form \( \Rightarrow \) \( t^V \) normal form

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME** \( t \rightarrow u \) \( \Rightarrow \) \( t^N \rightarrow u^N \)
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME**: $t$ normal form $\iff t^N$ normal form
- **VALUE**: $t$ normal form $\iff t^V$ normal form

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME**: $t \to u \implies t^N \to u^N$
Bang Calculus: A Subsuming Paradigm

**Static Properties:** [BucciarelliKesnerRíosViso’20,’23]

- **NAME**: $t$ normal form $\iff t^N$ normal form
- **VALUE**: $t$ normal form $\iff t^V$ normal form

**Dynamic Properties:** [BucciarelliKesnerRíosViso’20,’23]

- **NAME**: $t \mapsto u \implies t^N \mapsto u^N$
Bang Calculus: A Subsuming Paradigm

**Static Properties:** [BucciarelliKesnerRíosViso’20,’23]

- **NAME** \( t \) normal form \( \iff \) \( t^N \) normal form
- **VALUE** \( t \) normal form \( \iff \) \( t^V \) normal form

**Dynamic Properties:** [BucciarelliKesnerRíosViso’20,’23]

- **NAME** \( t \to u \) \( \Rightarrow \) \( t^N \to u^N \)
- **VALUE** \( t \to u \)
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME** \( t \text{ normal form} \) \( \iff \) \( t^N \text{ normal form} \)
- **VALUE** \( t \text{ normal form} \) \( \iff \) \( t^V \text{ normal form} \)

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME** \( t \rightarrow u \) \( \Rightarrow \) \( t^N \rightarrow u^N \)
- **VALUE** \( t \rightarrow u \)
Victor Arrial
Giulio Guerrieri
Delia Kesner

Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

NAME
$ t $ normal form $ \iff t^N $ normal form
VALUE
$ t $ normal form $ \iff t^V $ normal form

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

NAME
$ t \rightarrow u $ $ \iff t^N \rightarrow u^N $
VALUE
$ t \rightarrow u $ $ \iff t^V \rightarrow u^V $
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME**: $t$ normal form $\iff t^N$ normal form
- **VALUE**: $t$ normal form $\iff t^V$ normal form

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

- **NAME**: $t \rightarrow u \Rightarrow t^N \rightarrow u^N$
- **VALUE**: $t \rightarrow u \Rightarrow t^V \rightarrow u^V$
Bang Calculus: A Subsuming Paradigm

Static Properties: [BucciarelliKesnerRíosViso’20,’23]

NAME $t$ normal form $\Leftrightarrow$ $t^N$ normal form
VALUE $t$ normal form $\Leftrightarrow$ $t^V$ normal form

Dynamic Properties: [BucciarelliKesnerRíosViso’20,’23]

NAME $t \rightarrow u$ $\Rightarrow$ $t^N \rightarrow u^N$
VALUE $t \rightarrow u$ $\Rightarrow$ $t^V \rightarrow u^V$
Exploring the Bang-Calculus and Its Embeddings
Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework
& The Benefits of Diligence

Victor Arrial\textsuperscript{1}  Giulio Guerrieri\textsuperscript{2}  Delia Kesner\textsuperscript{1}

\textsuperscript{1}Université Paris Cité, Paris  \textsuperscript{2}Aix Marseille Univ, Marseille

LoVe Seminars
Université de Villetaneuse, November 30, 2023
Exploring the Bang-Calculus and Its Embeddings
Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework
& The Benefits of Diligence

Victor Arrial\textsuperscript{1}  Giulio Guerrieri\textsuperscript{2}  Delia Kesner\textsuperscript{1}

\textsuperscript{1}Université Paris Cité, Paris  \textsuperscript{2}Aix Marseille Univ, Marseille

LoVe Seminars
Université de Villetaneuse, November 30, 2023
Exploring the Bang-Calculus and Its Embeddings

Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework
& The Benefits of Diligence

Victor Arrial\textsuperscript{1}    Giulio Guerrieri\textsuperscript{2}    Delia Kesner\textsuperscript{1}

\textsuperscript{1}Université Paris Cité, Paris    \textsuperscript{2}Aix Marseille Univ, Marseille

LoVe Seminars
Université de Villetaneuse, November 30, 2023
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \]

Untyped terms

Terminating terms

Typable terms

[CoDe'78], [CoDe'80], [Gard'94], [Kfou'00]

Qualitative properties

Quantitative properties

[dCarv'07]
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \]
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \]

Untyped terms

Terminating terms

Typable terms
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:
\[ A \cap A = A \neq A \cap A \]
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B \]

Untyped terms

Terminating terms

Typable terms
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B \]

Untyped terms

Terminating terms

Typable terms
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:
\[
A \cap A = A
\]

Qualitative properties

Quantitative properties

[dCarv'07], [CoDe'78], [CoDe'80], [Gard'94], [Kfou'00]
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \mid A \cap B \]

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

Untyped terms

Terminating terms

= Tympable 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

\[ A, B ::= \sigma \mid A \Rightarrow B \mid A \land B \]

- **Untyped terms**
  - Terminating terms
    - Typable terms

- **Associativity:**
  \[ A \land (B \land C) = (A \land B) \land C \]

- **Commutativity:**
  \[ A \land B = B \land A \]

- **Idempotency?**

---

**Qualitative properties**

- Untyped terms
- Terminating terms
- Typable terms

**Quantitative properties**

- Associativity
- Commutativity
- 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**

[CoDe’78], [CoDe’80]

\[ A \cap A = A \]
Simple Types Versus Intersection Types

\[ A, B ::= \sigma \mid A \Rightarrow B \mid A \land B \]

- **Untyped terms**
- **Terminating terms**
- **Typable terms**

<table>
<thead>
<tr>
<th>Qualitative properties</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Idempotent</strong></td>
</tr>
<tr>
<td>[CoDe’78],[CoDe’80]</td>
</tr>
<tr>
<td>[ A \land A = A ]</td>
</tr>
</tbody>
</table>

- **Associativity:**
  \[ A \land (B \land C) = (A \land B) \land C \]

- **Commutativity:**
  \[ A \land B = B \land A \]

- **Idempotency?**

<table>
<thead>
<tr>
<th>Idempotent</th>
</tr>
</thead>
<tbody>
<tr>
<td>[CoDe’78],[CoDe’80]</td>
</tr>
<tr>
<td>[ A \land A = A ]</td>
</tr>
</tbody>
</table>
Simple Types Versus Intersection Types

\[ A, B ::= \sigma | A \Rightarrow B | A \cap B \]

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

- **Commutativity:**
  \[ A \cap B = B \cap A \]

- **Idempotency?**

<table>
<thead>
<tr>
<th>Idempotent</th>
<th>Non-Idempotent</th>
</tr>
</thead>
<tbody>
<tr>
<td>([\text{CoDe}'78],[\text{CoDe}'80])</td>
<td>([\text{Gard}'94], [\text{Kfou}'00])</td>
</tr>
<tr>
<td>[ A \cap A = A ]</td>
<td>[ A \cap A \neq A ]</td>
</tr>
</tbody>
</table>

**Qualitative properties**

- [✓]
- [✗]
Simple Types Versus Intersection Types

\[ A, B ::= \sigma | A \Rightarrow B \mid A \cap B \]

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?**

<table>
<thead>
<tr>
<th>Idempotent</th>
<th>Non-Idempotent</th>
</tr>
</thead>
<tbody>
<tr>
<td>[CoDe’78], [CoDe’80]</td>
<td>[Gard’94], [Kfou’00]</td>
</tr>
<tr>
<td>( A \cap A = A )</td>
<td>( A \cap A \neq A )</td>
</tr>
</tbody>
</table>

Qualitative properties

| ✔️ | ✗ |

Quantitative properties

[dCarv’07]
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Typability</th>
<th>Inhabitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Types</td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td>Idempotent Types</td>
<td>Indecidable</td>
<td>Indecidable</td>
</tr>
<tr>
<td>[Urz’99]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-Idempotent Types</td>
<td>Indecidable</td>
<td></td>
</tr>
<tr>
<td>(CBV)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(CBN) Decidable</td>
<td></td>
<td></td>
</tr>
<tr>
<td>[BKR’18]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th>Typing</th>
<th>Inhabitation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$? \vdash t : ?$</td>
<td>$\Gamma \vdash ? : \sigma$</td>
</tr>
</tbody>
</table>

- Simple Types: Decidable
- Idempotent Types: Decidable
- Non-Idempotent Types: Indecidable
- (CBN) - Decidable
- (CBV) - ?
<table>
<thead>
<tr>
<th></th>
<th>Typing</th>
<th>Inhabitation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$\Gamma \vdash ? : \sigma$</td>
<td></td>
</tr>
<tr>
<td>Simple Types</td>
<td>$? \vdash t : ?$</td>
<td></td>
</tr>
<tr>
<td>Idempotent Types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-Idempotent Types</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th></th>
<th>Typing $\Gamma \vdash ? : \sigma$</th>
<th>Inhabitation $\Gamma \vdash ? : \sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Types</td>
<td></td>
<td>Decidable</td>
</tr>
<tr>
<td>Idempotent Types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Non-Idempotent Types</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th></th>
<th>Typing $\vdash t : ?$</th>
<th>Inhabitation $\Gamma \vdash ? : \sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Types</td>
<td>Decidable</td>
<td></td>
</tr>
<tr>
<td>Idempotent Types</td>
<td>Decidable</td>
<td>Indecidable</td>
</tr>
<tr>
<td>Non-Idempotent Types</td>
<td>Indecidable</td>
<td>Indecidable</td>
</tr>
</tbody>
</table>

- Decidable: Typing and Inhabitation are decidable
- Indecidable: Typing and Inhabitation are undecidable
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th></th>
<th>Typing $\Gamma \vdash \tau \colon \tau$</th>
<th>Inhabitation $\Gamma \vdash ? : \sigma$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Simple Types</strong></td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td><strong>Idempotent Types</strong></td>
<td>Indecidable</td>
<td>Indecidable</td>
</tr>
<tr>
<td><strong>Non-Idempotent Types</strong></td>
<td>Indecidable</td>
<td>Indecidable</td>
</tr>
</tbody>
</table>
Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th></th>
<th>Typing  ( ? \vdash t : ? )</th>
<th>Inhabitation ( \Gamma \vdash ? : \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Types</td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td>Idempotent Types</td>
<td>Indecidable</td>
<td>Indecidable [Urz’99]</td>
</tr>
<tr>
<td>Non-Idempotent Types</td>
<td>Indecidable</td>
<td></td>
</tr>
</tbody>
</table>

Simple Types: typing is decidable and inhabitation is decidable.
Idempotent Types: typing is decidable, but inhabitation is indecidable and \[Urz’99\] indicates a specific reference.
Non-Idempotent Types: both typing and inhabitation are indecidable.
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Typing ( ? \vdash t : ? )</th>
<th>Inhabitation ( \Gamma \vdash ? : \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Types</td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td>Idempotent Types</td>
<td>Indecidable</td>
<td>Indecidable ([\text{Urz'99}])</td>
</tr>
<tr>
<td>Non-Idempotent Types</td>
<td>Indecidable</td>
<td>(CBN) Decidable ([\text{BKR'18}])</td>
</tr>
<tr>
<td></td>
<td>Typing (? \vdash t : , ?)</td>
<td>Inhabitation (? \vdash , ? : , \sigma)</td>
</tr>
<tr>
<td>----------------------</td>
<td>-------------------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td><strong>Simple Types</strong></td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td><strong>Idempotent Types</strong></td>
<td>Indecidable</td>
<td>Indecidable [Urz’99]</td>
</tr>
<tr>
<td><strong>Non-Idempotent Types</strong></td>
<td>Indecidable</td>
<td>(CBN) Decidable [BKR’18]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(CBV) ?</td>
</tr>
</tbody>
</table>
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th></th>
<th>Typing ( \text{?} \vdash t : \text{?} )</th>
<th>Inhabitation ( \Gamma \vdash \text{?} : \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Simple Types</strong></td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td><strong>Idempotent Types</strong></td>
<td>Indecidable</td>
<td>Indecidable [Urz’99]</td>
</tr>
<tr>
<td><strong>Non-Idempotent Types</strong></td>
<td>Indecidable</td>
<td>(CBN) Decidable [BKR’18]</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(CBV) ?</td>
</tr>
</tbody>
</table>
## Typability and Inhabitation in Intersection Types

<table>
<thead>
<tr>
<th></th>
<th>Typing ( ? \vdash t : ? )</th>
<th>Inhabitation ( \Gamma \vdash ? : \sigma )</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Simple Types</strong></td>
<td>Decidable</td>
<td>Decidable</td>
</tr>
<tr>
<td><strong>Idempotent Types</strong></td>
<td>Indecidable</td>
<td>Indecidable [Urz’99]</td>
</tr>
<tr>
<td><strong>Non-Idempotent Types</strong></td>
<td>Indecidable</td>
<td>(CBN) Decidable [BKR’18] (CBV) Decidable</td>
</tr>
</tbody>
</table>
Intersection Types and Distant Bang Calculus

Three Typing Systems:

- \( N:V \) - Static Properties

\[ N:V \]

\[ V:V \]
Intersection Types and Distant Bang Calculus

Three Typing Systems: [BKRV’20]

\[
\text{NAME} : N \quad \text{VALUE} : V \quad \text{BANG} : B
\]
Intersection Types and Distant Bang Calculus

Three Typing Systems: [BKRV’20]

\[
\begin{align*}
\text{NAME} & : \mathcal{N} & \text{VALUE} & : \mathcal{V} & \text{BANG} & : \mathcal{B}
\end{align*}
\]

Static Properties: [BKRV’20]

\[
\Gamma \vdash_N t : \sigma
\]
Three Typing Systems: [BKRV'20]

- **NAME**: $N$
- **VALUE**: $V$
- **BANG**: $B$

Static Properties: [BKRV'20]

\[
\Gamma \vdash_N t : \sigma \quad \iff \quad \Gamma \vdash_B t^N : \sigma
\]
Intersection Types and Distant Bang Calculus

Three Typing Systems: [BKRV’20]

- NAME : N
- VALUE : V
- BANG : B

Static Properties: [BKRV’20]

- NAME: $\Gamma \vdash_N t : \sigma \iff \Gamma \vdash_B t^N : \sigma$
- VALUE: $\Gamma \vdash_V t : \sigma \iff \Gamma \vdash_B t^V : \sigma$
Exploring the Bang-Calculus and Its Embeddings
Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework
& The Benefits of Diligence

Victor Arrial¹
Giulio Guerrieri²
Delia Kesner¹

¹ Université Paris Cité, Paris ² Aix Marseille Univ, Marseille

LoVe Seminars
Université de Villetaneuse, November 30, 2023
Exploring the Bang-Calculus and Its Embeddings
Quantitative Inhabitation for Different Lambda Calculi in a Unifying Framework
& The Benefits of Diligence

Victor Arrial\textsuperscript{1} Giulo Guerrieri\textsuperscript{2} Delia Kesner\textsuperscript{1}

\textsuperscript{1}Université Paris Cité, Paris \quad \textsuperscript{2}Aix Marseille Univ, Marseille

LoVe Seminars
Université de Villetaneuse, November 30, 2023
Coming Back to Inhabitation

- First Goal: Decidability of the (more general) Inhabitation Problem (IP).

- More Ambitious Second Goal: Decidability of the and IP from decidability of the IP.

- More Ambitious Third Goal: Decidability by finding all inhabitants in the IP. Decidability of the and IP by finding all inhabitants from those of the IP. Using generic properties so that other encodable models of computation can use these results.
Coming Back to Inhabitation

First Goal

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

Using generic properties so that other encodable models of computation can use these results.
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.
Instead of just one solution:

\[ t : \text{Sol}(\Gamma, M) \]

We want to compute all solutions:

\[ \text{Sol}(\Gamma, M) = \{ t | t : \text{Sol}(\Gamma, M) \} \]

The set \( \text{Sol}(\Gamma, M) \) is either empty or infinite.

We compute a finite generator:

\[ \text{Basis}(\Gamma, M) \]

Which is correct and complete:

\[ \text{span}(\text{Basis}(\Gamma, M)) = \text{Sol}(\Gamma, M) \]

Theorem

For any typing \((\Gamma, M)\), \(\text{Basis}(\Gamma, M)\) exists, is finite, correct and complete.
Instead of just one solution:
\[ \Gamma \vdash t : \sigma \]
We want to compute all solutions:
\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]
Instead of just one solution:

\[ \Gamma \vdash t : \sigma \]

We want to compute all solutions:

\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]
Instead of just one solution:
\[ \Gamma \vdash t : \sigma \]
We want to compute all solutions:
\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]

Problem

✗ The set \( \text{Sol}(\Gamma, \sigma) \) is either empty or infinite

\[ \text{BANG} \]
Instead of just one solution:
\[ \Gamma \vdash t : \sigma \]
We want to compute all solutions:
\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]

Problem

✗ The set \( \text{Sol}(\Gamma, \sigma) \) is either empty or infinite

We compute a finite generator:
\[ \text{Basis}(\Gamma, \sigma) \]
Which is correct and complete:
\[ \text{span}(\text{Basis}(\Gamma, \sigma)) = \text{Sol}(\Gamma, \sigma) \]
Instead of just one solution:
\[ \Gamma \vdash t : \sigma \]
We want to compute all solutions:
\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]

**Problem**

× The set \( \text{Sol}(\Gamma, \sigma) \) is either empty or infinite

We compute a finite generator:
\[ \text{Basis}(\Gamma, \sigma) \]
Which is correct and complete:
\[ \text{span}(\text{Basis}(\Gamma, \sigma)) = \text{Sol}(\Gamma, \sigma) \]
Instead of just one solution:
\[ \Gamma \vdash t : \sigma \]
We want to compute all solutions:
\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]

Problem

\[ \text{The set Sol}(\Gamma, \sigma) \text{ is either empty or infinite} \]

We compute a finite generator:
\[ \text{Basis}(\Gamma, \sigma) \]
Which is correct and complete:
\[ \text{span}(\text{Basis}(\Gamma, \sigma)) = \text{Sol}(\Gamma, \sigma) \]
Instead of \textbf{just one} solution:
\[
\Gamma \vdash t : \sigma
\]
We want to compute \textbf{all} solutions:
\[
\text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \}
\]

**Problem**

\text{✗} The set \text{Sol}(\Gamma, \sigma) is either empty or infinite

We compute a \textbf{finite} generator:
\[
\text{Basis}(\Gamma, \sigma)
\]
Which is \textbf{correct} and \textbf{complete}:
\[
\text{span}(\text{Basis}(\Gamma, \sigma)) = \text{Sol}(\Gamma, \sigma)
\]

**Theorem**

\text{✓} For any typing \((\Gamma, \sigma)\), \text{Basis}_{\mathcal{B}}(\Gamma, \sigma) \textbf{exists}, is \textbf{finite}, \textbf{correct} and \textbf{complete}. 
Instead of just one solution:

\[ \Gamma \vdash t : \sigma \]

We want to compute all solutions:

\[ \text{Sol}(\Gamma, \sigma) := \{ t \mid \Gamma \vdash t : \sigma \} \]

**Problem**

- The set \( \text{Sol}(\Gamma, \sigma) \) is either empty or infinite

**Theorem**

- For any typing \((\Gamma, \sigma)\), \( \text{Basis}_{\mathcal{G}}(\Gamma, \sigma) \) exists, is finite, correct and complete.
Following the Typing and a Grammar

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

Sol \((\cdot, \cdot)\)

Basis

B
Computing the basis:
Recreate typing trees, but only on elements of the Basis.
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.

Sol(Γ, σ)

Follows two sets of rules:
- Typing rules
Computing the basis:
Recreate typing trees, but only on elements of the Basis.

Follows two sets of rules:
- Typing rules
- Grammar rules
Computing the basis:
Recreate typing trees, but only on elements of the Basis.

Follows two sets of rules:
- Typing rules
- Grammar rules
The Full Algorithm
The Full Algorithm
The Full Algorithm
The Full Algorithm
The Full Algorithm
The Full Algorithm

\[
g \leadsto \text{App}(g_a, g_b)
\]

\[
\Gamma = \Gamma_a + \Gamma_b
\]

\[
\mathcal{M} \Rightarrow \sigma \vdash S(\tau, \Diamond \Rightarrow \sigma)
\]

\[
a \vdash_{g_a} H^x[\tau](\Gamma_a; \mathcal{M} \Rightarrow \sigma)
\]

\[
b \vdash_{g_b} N(\Gamma_b; \mathcal{M})
\]

\[
ab \vdash_{g} H^x[\tau](\Gamma; \sigma)
\]
The Full Algorithm
The Full Algorithm and its Implementation

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

github/ArrialVictor/InhabitationLambdaBang
Properties of the Inhabitation Algorithm

Theorem

- The inhabitation algorithm terminates.
- The algorithm is sound and complete (i.e. it exactly computes $B$).

More Ambitious Third Goal

Decidability by finding all inhabitants in the IP.

Decidability of the and IP by finding all inhabitants from those of the IP.

Using generic properties so that other encodable models of computation can use these results.
Properties of the Inhabitation Algorithm

Non-deterministic algorithm
Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Theorem

✔ The inhabitation algorithm terminates.
Non-deterministic algorithm

**Theorem**

- ✔ *The inhabitation algorithm terminates.*
- ✔ *The algorithm is sound and complete (i.e. it exactly computes Basis_B(Γ, σ)).*
Properties of the Inhabitation Algorithm

Non-deterministic algorithm

Theorem

- The inhabitation algorithm terminates.
- The algorithm is sound and complete (i.e. it exactly computes Basis_{B}(\Gamma, \sigma)).

More Ambitious Third Goal

- Decidability by finding all inhabitants in the 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 \( \text{Basis}_B(\Gamma, \sigma) \)).**

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.
For any typing \((\tau, \Delta)\), Basis \(N(\tau, \Delta)\) exists, is finite, correct and complete. Built an algorithm computing Basis \(N(\tau, \Delta)\): [BKR'14]
Theorem ([BKR’14])

For any typing \((\Gamma, \sigma)\), \(\text{Basis}_N(\Gamma, \sigma)\) exists, is finite, correct and complete.
Theorem ([BKR’14])

For any typing \((\Gamma, \sigma)\), \(\text{Basis}_N(\Gamma, \sigma)\) exists, is finite, correct and complete.

Built an algorithm computing \(\text{Basis}_N(\Gamma, \sigma)\): [BKR’14]

\[
\begin{align*}
\frac{a \vdash T(\Gamma + x : A, \tau) \quad x \notin \text{dom}(\Gamma)}{\lambda x.a \vdash T(\Lambda, A \to \tau)} \text{ (Abs)} \\
\frac{(a_i \vdash T(\Gamma_i, \sigma_i))_{i \in I} \quad \bigcup_{i \in I} a_i}{\bigcup_{i \in I} a_i \vdash \text{TI}(\bigcup_{i \in I} \Gamma_i, [\sigma_i]_{i \in I})} \text{ (Union)} \\
\frac{\Gamma = \Gamma_1 + \Gamma_2 \quad a \vdash H^{\xi;[\Gamma_1 \to \cdots \to B \to \tau]}(\Gamma_1, B \to \tau) \quad b \vdash \text{TI}(\Gamma_2, B) \quad n \geq 0}{ab \vdash H^{\xi;[\Gamma_1 \to \cdots \to B \to \tau]}(\Gamma, \tau)} \text{ (Head>0)} \\
\frac{x \vdash H^{\xi;[\tau]}(\emptyset, \tau)}{x \vdash H^{\xi;[\tau]}(\emptyset, \tau)} \text{ (Head0)} \\
\frac{a \vdash H^{\xi;[A_1 \to \cdots \to A_n \to \tau]}(\Gamma, \tau)}{a \vdash T(\Gamma + x : [A_1 \to \cdots \to A_n \to \tau], \tau)} \text{ (Head)}
\end{align*}
\]
Solving Inhabitation: through Inhabitation
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma,\sigma) \]
The Basis is preserved by the embedding:

Theorem

$$\text{NAME} \quad t \in \text{Basis}_N(\Gamma, \sigma) \quad \iff \quad t^N \in \text{Basis}_B(\Gamma, \sigma) \quad \text{BANG}$$
The Basis is preserved by the embedding:

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

\[
\text{Theorem}\quad t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma)
\]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_N(\Gamma, \sigma) \iff t^N \in \text{Basis}_B(\Gamma, \sigma) \]
Solving Inhabitation - Usual Methology

For any typing \((\tau_1, \tau_2)\), basis \(V(\tau_1, \tau_2)\) exists, is finite, correct and complete. Built an algorithm computing \(V(\tau_1, \tau_2)\):
Theorem

For any typing \((\Gamma, \sigma)\), \(\text{Basis}_V(\Gamma, \sigma)\) exists, is finite, correct and complete.
Theorem

For any typing \((\Gamma, \sigma)\), \(Basis_\mathcal{V}(\Gamma, \sigma)\) exists, is finite, correct and complete.

Built an algorithm computing \(Basis_\mathcal{V}(\Gamma, \sigma)\):
Theorem

For any typing $(\Gamma, \sigma)$, \( \text{Basis}_{\mathcal{V}}(\Gamma, \sigma) \) exists, is finite, correct and complete.

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

\[
\begin{align*}
\Gamma = \Gamma_1 + \Gamma_2, & \quad \text{fix } x \notin \text{dom}(\Gamma) \cup \{x\} \\
\quad n \in [0, \text{sz}(\rho)], & \quad \text{fix } y \notin \text{dom}(\Gamma) \cup \{x\} \\
\quad j \in [1, n], & \quad a \vDash S(\rho_j, \sigma) \\
\Gamma = \Gamma_1 + \Gamma_2, & \quad \text{fix } x \notin \text{dom}(\Gamma) \cup \{x\} \\
\quad n \in [0, \text{sz}(\tau)], & \quad \text{fix } y \notin \text{dom}(\Gamma) \cup \{x\} \\
\quad j \in [1, n], & \quad a \vDash S(\tau_j, \sigma) \cup \sigma \\
\Gamma = \Gamma_1 + \Gamma_2, & \quad \text{fix } x \notin \text{dom}(\Gamma) \\
\quad N \in [0, \text{sz}(\tau)], & \quad \text{fix } y \notin \text{dom}(\Gamma) \\
\quad j \in [1, n], & \quad a \vDash S(\rho_j, \sigma) \\
\end{align*}
\]
Solving\textbf{VALUE} Inhabitation : through \textbf{BANG} Inhabitation
The Basis is preserved by the embedding:

Theorem

\[ t \in \text{Basis}_V(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

$$\text{Theorem}$$

$$t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma)$$
The Basis is preserved by the embedding:

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[
t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma)
\]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_{V}(\Gamma, \sigma) \iff t^V \in \text{Basis}_{B}(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

Theorem

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

Theorem

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

\[ t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \]
The Basis is preserved by the embedding:

**Theorem**

\( t \in \text{Basis}_V(\Gamma, \sigma) \iff t^V \in \text{Basis}_B(\Gamma, \sigma) \)
Properties of the Indirect **NAME** and **VALUE** Algorithm

The inhabitation algorithm terminates. The algorithm is sound and complete (i.e., it exactly computes Basis $B(x, y)$).

**More Ambitious Third Goal**

Decidability by finding all inhabitants in the IP.

Decidability of the and IP by finding all inhabitants from those of the IP.

Using generic properties so that other encodable models of computation can use these results.
Theorem

- The inhabitation algorithm terminates.
- The algorithm is sound and complete (i.e. it exactly computes $\text{Basis}_G(\Gamma, \sigma)$).
Properties of the Indirect Algorithm

**Theorem**

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

**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.
### Theorem

- The inhabitation algorithm terminates.
- The algorithm is sound and complete (i.e., it exactly computes 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 Algorithm

**Theorem**

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

**More Ambitious Third Goal**

- Decidability by finding all inhabitants in the IP.
- Decidability of the IP by finding all inhabitants from those of the 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 \((\Gamma, \sigma)\)).

**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.
Theorem

- The inhabitation algorithm terminates.
- The algorithm is sound and complete (i.e. it exactly computes 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 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.