

# 必要呼び意味論と名前呼び意味論の 対応の形式的検証

水野 雅之

情報科学研究科 情報基礎科学専攻  
住井・松田研究室

指導教員：住井 英二郎 教授

## 動機：非正格言語の仕様と実装のギャップ

- 名前呼び：非正格言語の(ハイレベルな)仕様
- 必要呼び：非正格言語の実装

これらの対応を検証したい

## $\lambda$ 計算の基本 : full- $\beta$ 簡約

### ■ 簡約基は複数存在しうる

- $(\lambda xy. y) ((\lambda x. xx) (\lambda x. xx))$
- $(\lambda xy. y) ((\lambda x. xx) (\lambda x. xx))$

### ■ 下手に簡約基を選ぶと無限ループに

- $(\lambda xy. y) ((\lambda x. xx) (\lambda x. xx))$   
 $\xrightarrow{\beta} \lambda y. y$
- $(\lambda xy. y) ((\lambda x. xx) (\lambda x. xx))$   
 $\xrightarrow{\beta} (\lambda xy. y) ((\lambda x. xx) (\lambda x. xx))$

# 非正格な言語の理論的背景：名前呼び

## 定義 (名前呼び)

関数呼び出し以前に引数を簡約せず，  
 $\lambda$  抽象の中も簡約しない評価戦略

- full- $\beta$  簡約で  $\lambda$  抽象に評価できる項は名前呼びでも  $\lambda$  抽象まで評価できる

$$\underline{(\lambda xy. x) ((\lambda x. xx) (\lambda x. xx))} \xrightarrow{\text{name}} \lambda y. y$$

### 関数引数の簡約基が複製される

- 余計な簡約が必要になることも

- $(\lambda x. xx) \underbrace{((\lambda x. x) (\lambda x. x))}_{\beta}$

$$\xrightarrow{\beta} (\lambda x. xx) (\lambda x. x)$$

- $(\lambda x. xx) \underbrace{((\lambda x. x) (\lambda x. x))}_{\text{name}}$

$$\xrightarrow{\text{name}}$$

$$(\lambda x. x) (\lambda x. x) ((\lambda x. x) (\lambda x. x))$$

## 名前呼びの改良：必要呼び

- 一度関数引数を評価したら値を覚えて使い回す

- 余計な簡約が不要に

$$(\lambda x. xx) ((\lambda x. x) (\lambda x. x))$$
$$\xrightarrow{\text{need}} \text{let } x = (\lambda x. x) (\lambda x. x) \text{ in } x x$$
$$\xrightarrow{\text{need}} \text{let } x = \lambda x. x \text{ in } x x$$

- 名前呼びと振る舞いが一致してほしい

# 本研究の貢献

- 必要呼び  $\lambda$  計算 [Ariola+ 1995] を検証に適するよう変形し Coq で定式化
  - De Bruijn インデックス
  - 評価文脈の排除
- 名前呼びとの評価の対応に既存研究より簡単な証明を考案, Coq で定式化
  - 標準化定理 [Curry&Feys 1958] を利用

# アウトライン

- ① 対象言語：名前呼び  $\lambda$  計算と必要呼び  $\lambda$  計算
- ② 貢献 1：主定理の簡潔化された証明
- ③ 貢献 2：必要呼びラムダ計算の形式的検証に適した定式化
- ④ まとめ



# アウトライン

- ① 対象言語：名前呼び  $\lambda$  計算と必要呼び  $\lambda$  計算
- ② 貢献 1：主定理の簡潔化された証明
- ③ 貢献 2：必要呼びラムダ計算の形式的検証に適した定式化
- ④ まとめ

# 名前呼び $\lambda$ 計算

Terms	$L, M, N$	$::=$	$x \mid V \mid M \ N$
Values (WHNF)	$V$	$::=$	$\lambda x.M$
Evaluation contexts	$E_n$	$::=$	$[] \mid E_n \ M$

$$(\beta) \quad (\lambda x.M)N \rightarrow M[x \mapsto N]$$

- 簡約は決定的
- stuck 状態は全て  $E_n[x]$  の形

## 補題 (名前呼び $\lambda$ 計算の決定性)

- $\xrightarrow{\text{name}}$  は部分関数
- $E_n[x] = E'_n[y]$  ならば  $x = y$
- 任意の  $M$  について, 以下のうちただ一つが成り立つ
  1.  $M$  は値
  2.  $M = E_n[x]$  となる  $E_n$  と  $x$  が存在する
  3.  $M$  は  $\xrightarrow{\text{name}}$  で簡約できる

# 標準化定理 [Curry&Feys 1958]

## 定義 (標準簡約列)

$i > j$  であるような全ての  $i$  と  $j$  について, 簡約位置  $\Delta_i$  が  $\Delta_j$  よりも外側かつ左側にある場合, 簡約列

$M_1 \xrightarrow[\Delta_1]{\beta} M_2 \xrightarrow[\Delta_2]{\beta} \cdots \xrightarrow[\Delta_{n-1}]{\beta} M_n$  を標準簡約列という

## 定理 (標準化定理)

$M \xrightarrow{\beta} N$  ならば,  $M$  から  $N$  への標準簡約列が存在する

# 名前呼びの性質：標準化定理の2つの系

## 定理 (正規化定理)

$M$  は full- $\beta$  簡約で  $\lambda$  抽象に評価できるならば,  $M$  は 名前呼びの簡約でも  $\lambda$  抽象に評価できる

## 定理 (準正規化定理)

$M$  は full- $\beta$  簡約で  $\lambda$  抽象に評価できるならば,  
 $\xrightarrow{\text{name}} \circ \xrightarrow{\beta}$  による  $M$  の簡約は必ず止まる

- 必要呼びとの対応の証明に用いる

# 必要呼び $\lambda$ 計算 [Ariola+ 1995]

Terms	$M, N ::= x \mid V \mid M N \mid \text{let } x = M \text{ in } N$
Values	$V ::= \lambda x. M$
Answers	$A ::= V \mid \text{let } x = M \text{ in } A$
Evaluation contexts	$E, E' ::= [] \mid E M \mid \text{let } x = M \text{ in } E$ $\mid \text{let } x = E \text{ in } E'[x]$

- (I)  $(\lambda x. M)N \rightarrow \text{let } x = N \text{ in } M$   
(V)  $\text{let } x = V \text{ in } E[x] \rightarrow \text{let } x = V \text{ in } E[V]$   
(C)  $(\text{let } x = M \text{ in } A) N \rightarrow \text{let } x = M \text{ in } A N$   
(A)  $\text{let } y = (\text{let } x = M \text{ in } A) \text{ in } E[y] \rightarrow$   
 $\text{let } x = M \text{ in let } y = A \text{ in } E[y]$

$\xrightarrow{\text{I}}$  簡約規則 (I) のみ用いる簡約

$\xrightarrow{\text{VCA}}$  簡約規則 (V), (C) と (A) のみ用いる簡約

## 補題 (必要呼び $\lambda$ 計算の決定性)

- $\overset{I}{\rightarrow}$  は部分関数
- $\overset{VCA}{\longrightarrow}$  は部分関数
- $E[x] = E'[y]$  ならば  $x = y$
- 任意の  $M$  について, 以下のうちただ一つが成り立つ
  1.  $M$  は answer
  2.  $M = E[x]$  となる  $E$  と  $x$  が存在する
  3.  $M$  は  $\overset{I}{\rightarrow}$  で簡約できる
  4.  $M$  は  $\overset{VCA}{\longrightarrow}$  で簡約できる

# アウトライン

- ① 対象言語：名前呼び  $\lambda$  計算と必要呼び  $\lambda$  計算
- ② 貢献 1：主定理の簡潔化された証明
- ③ 貢献 2：必要呼びラムダ計算の形式的検証に適した定式化
- ④ まとめ



# 主定理：名前呼びと必要呼びの対応

## 定理（必要呼び評価の健全性）

$M$  が必要呼びの簡約によって  $\text{answer}$  に評価できるならば、 $M$  に対応する項  $N$  は名前呼びの簡約によって  $\lambda$  抽象に評価される

## 定理（必要呼び評価の完全性）

$M$  が名前呼びの簡約によって  $\lambda$  抽象に評価できるならば、 $M$  に対応する項  $N$  は必要呼びの簡約によって  $\text{answer}$  に評価される

# 先行研究の証明方法

- [Ariola+ 1995] は Ariola らの研究と Maraist らの研究を合わせたもの

—— Ariola と Felleisen[1997] の証明 ——

- Informal に定義されたグラフに基づく

—— Maraist ら [1998] の証明 ——

- マーク付きの簡約と簡約位置を陽に扱うのが煩雑
- 本研究で一部参考

## 定義 (項同士の対応関係)

$$x^{\dagger} = x$$

$$(\lambda x.M)^{\dagger} = \lambda x.M^{\dagger}$$

$$(M\ N)^{\dagger} = M^{\dagger}\ N^{\dagger}$$

$$(\text{let } x = M \text{ in } N)^{\dagger} = N^{\dagger}[x \mapsto M^{\dagger}]$$

- let を全て展開する関数
- マークを除くと [Maraist+ 1998] と一致

## 補題 (1 ステップの対応)

- $A^\flat$  は値
- 任意の  $E$  と  $x$  について,  $E[x]^\flat = E_n[x]$  となる  $E_n$  が存在
- $M \xrightarrow{\text{VCA}} N$  ならば  $M^\flat = N^\flat$
- $M \xrightarrow{\text{I}} N$  ならば  $M^\flat \xrightarrow{\text{name}} \circ \xrightarrow{\beta} N^\flat$

## 再掲：前呼びと必要呼びの評価の対応

### 定理（必要呼び評価の健全性）

$M \xrightarrow{\text{need}} A$  ならば,  $M^{\uparrow} \xrightarrow{\text{name}} V$  となるような値  $V$  が存在する

### 定理（必要呼び評価の完全性）

$M^{\uparrow} \xrightarrow{\text{name}} V$  ならば,  $M \xrightarrow{\text{need}} A$  となるような answer  $A$  が存在する

## 再掲：標準化定理の2つの系

### 定理 (正規化定理)

$M$  は full- $\beta$  簡約で  $\lambda$  抽象に評価できるならば,  $M$  は 名前呼びの簡約でも  $\lambda$  抽象に評価できる

### 定理 (準正規化定理)

$M$  は full- $\beta$  簡約で  $\lambda$  抽象に評価できるならば,  
 $\xrightarrow{\text{name}} \circ \xrightarrow{\beta}$  による  $M$  の簡約は必ず止まる

## 健全性の証明.

$$\begin{array}{ccc}
 M & \xrightarrow{\text{need}} & A \\
 \downarrow \text{\tiny \(\hookrightarrow\)} & & \downarrow \text{\tiny \(\hookrightarrow\)} \\
 M^{\text{\tiny \(\hookrightarrow\)}} & \xrightarrow{\text{name}} & V \\
 & \searrow \beta & \nearrow \\
 & & A^{\text{\tiny \(\hookrightarrow\)}}
 \end{array}$$

1ステップの対応により  $A^{\text{\tiny \(\hookrightarrow\)}}$  は値であるから, 正規化定理により  $M^{\text{\tiny \(\hookrightarrow\)}} \xrightarrow{\text{name}} V$  となるような値  $V$  が存在する .



## 完全性の証明の困難な点

- Administrative な簡約が止まらないかもしれない
  - $M \xrightarrow{\text{VCA}} N$  ならば  $M^{\flat} = N^{\flat}$
- let で共有されている部分がすべて簡約される
  - $M \xrightarrow{\text{I}} N$  ならば  $M^{\flat} \xrightarrow{\text{name}} \circ \xrightarrow{\beta} N^{\flat}$
  - $\lambda$  抽象の中も簡約されうる



$\xrightarrow{\text{VCA}}$  の正規化性の証明.

[Maraist+ 1998] の重み付けを変形したものにより証明

$$\begin{aligned}\|x\|_s &= s(x) \\ \|\lambda x.M\|_s &= \|M\|_{s \circ [x \mapsto 1]} \\ \|MN\|_s &= 2\|M\|_s + 2\|N\|_s \\ \|\text{let } x = M \text{ in } N\|_s &= 2\|M\|_s + \|N\|_{s \circ [x \mapsto 1 + \|M\|_s]}\end{aligned}$$

$M \xrightarrow{\text{VCA}} N$  ならば  $\|M\|_s > \|N\|_s$



## 完全性の証明 (1/2).

$M \stackrel{\text{name}}{\longrightarrow} V$  と仮定する .

まず  $\stackrel{\text{need}}{\longrightarrow}$  によって  $M$  が正規化されることを示す

$$M \xrightarrow{\text{need}} \text{---} \xrightarrow{\text{need}} N$$

## 完全性の証明 (1/2).

$M^{\text{name}} \longrightarrow V$  と仮定する .

まず  $\xrightarrow{\text{need}}$  によって  $M$  が正規化されることを示す

$\xrightarrow{\text{need}} = (\xrightarrow{\text{I}} \cup \xrightarrow{\text{VCA}})^* = (\xrightarrow{\text{VCA}} \circ \xrightarrow{\text{I}})^*$  が成り立つ

$$M \xrightarrow{\text{VCA}} \xrightarrow{\text{I}} \cdots \xrightarrow{\text{VCA}} \xrightarrow{\text{I}} N$$

## 完全性の証明 (1/2).

$M^{\Downarrow} \xrightarrow{\text{name}} V$  と仮定する .

まず  $\xrightarrow{\text{need}}$  によって  $M$  が正規化されることを示す

準正規化定理により,  $\xrightarrow{\text{name}} \circ \xrightarrow{\beta}$  による  $M^{\Downarrow}$  の簡約は停止する (＝導出に関する帰納法を使える)

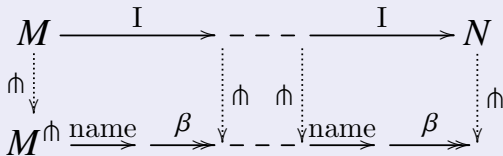
$$\begin{array}{ccccccc} M & \xrightarrow{\text{VCA}} & \xrightarrow{\text{I}} & \cdots & \xrightarrow{\text{VCA}} & \xrightarrow{\text{I}} & N \\ \Downarrow & & & & & & \\ M^{\Downarrow} & \xrightarrow{\text{name}} & \xrightarrow{\beta} & \cdots & \xrightarrow{\text{name}} & \xrightarrow{\beta} & \end{array}$$

## 完全性の証明 (1/2).

$M \Downarrow \xrightarrow{\text{name}} V$  と仮定する .

まず  $\xrightarrow{\text{need}}$  によって  $M$  が正規化されることを示す

$\xrightarrow{I}$  による簡約は  $\xrightarrow{\text{name}} \circ \xrightarrow{\beta}$  による簡約に対応 .



## 完全性の証明 (1/2).

$M \multimap^{\text{name}} \twoheadrightarrow V$  と仮定する .

まず  $\multimap^{\text{need}}$  によって  $M$  が正規化されることを示す

$\xrightarrow{\text{VCA}}$  は正規化性を満たす

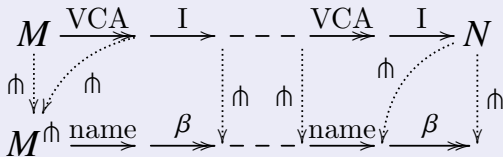
$$\begin{array}{ccccccc} M & \xrightarrow{\text{VCA}} & \xrightarrow{\text{I}} & \cdots & \xrightarrow{\text{VCA}} & \xrightarrow{\text{I}} & N \\ \downarrow \text{need} & & & & & & \\ M & \multimap^{\text{name}} \xrightarrow{\quad} & \xrightarrow{\beta} & \cdots & \xrightarrow{\text{name}} & \xrightarrow{\beta} & \end{array}$$

## 完全性の証明 (1/2).

$M \Downarrow^{\text{name}} \twoheadrightarrow V$  と仮定する .

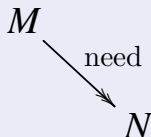
まず  $\xrightarrow{\text{need}}$  によって  $M$  が正規化されることを示す

$\xrightarrow{\text{VCA}}$  は administrative な簡約



## 完全性の証明 (2/2).

次に,  $M$  の正規形  $N$  が answer であることを示す

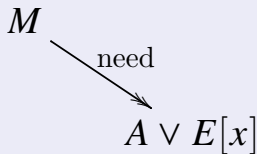




## 完全性の証明 (2/2).

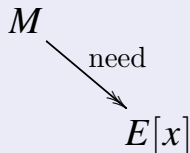
次に,  $M$  の正規形  $N$  が answer であることを示す

正規形  $N$  は answer または stuck 状態  $E[x]$  である



## 完全性の証明 (2/2).

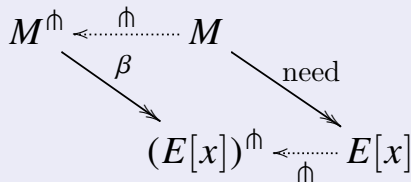
次に,  $M$  の正規形  $N$  が answer であることを示す  
正規形  $N$  が stuck 状態  $E[x]$  と仮定し, 矛盾を導く.



## 完全性の証明 (2/2).

次に,  $M$  の正規形  $N$  が answer であることを示す

1 ステップの対応により  $M^{\flat} \xrightarrow{\beta} (E[x])^{\flat}$



## 完全性の証明 (2/2).

次に,  $M$  の正規形  $N$  が answer であることを示す

1 ステップの対応により  $E[x]^{\flat} = E_n[x]$

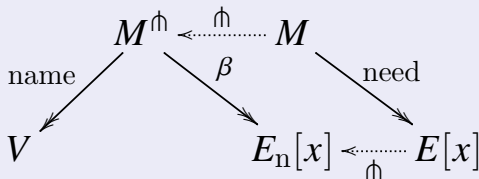
$$\begin{array}{ccc} M^{\flat} & \xleftarrow{\quad \flat \quad} & M \\ & \searrow \beta & \searrow \text{need} \\ & E_n[x] & \xleftarrow{\quad \flat \quad} E[x] \end{array}$$



## 完全性の証明 (2/2).

次に,  $M$  の正規形  $N$  が answer であることを示す

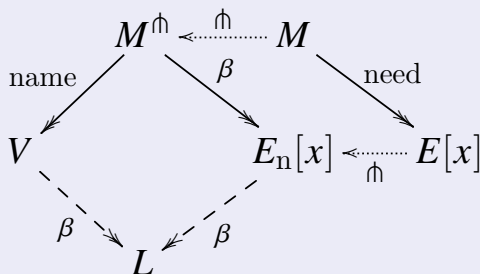
仮定より  $M^{\flat} \xrightarrow{\text{name}} V$



## 完全性の証明 (2/2).

次に,  $M$  の正規形  $N$  が answer であることを示す

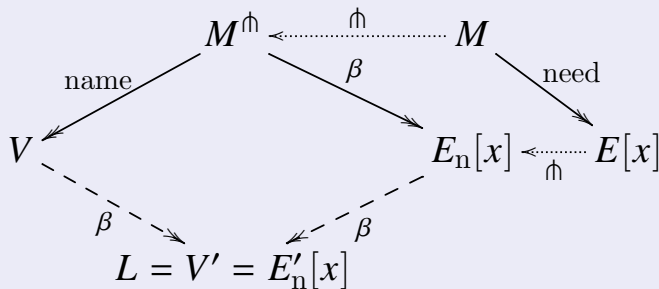
$\xrightarrow{\beta}$  の合流性により,  $V \xrightarrow{\beta} L$  かつ  $E_n[x] \xrightarrow{\beta} L$   
 となるような  $L$  が存在する ( $\xrightarrow{\text{name}} \subset \xrightarrow{\beta}$ )



## 完全性の証明 (2/2).

次に,  $M$  の正規形  $N$  が answer であることを示す

名前呼びにおける値であることや stuck 状態であることは, full- $\beta$  簡約で保存される



# アウトライン

- ① 対象言語：名前呼び $\lambda$ 計算と必要呼び $\lambda$ 計算
- ② 貢献 1：主定理の簡潔化された証明
- ③ 貢献 2：必要呼びラムダ計算の形式的検証に適した定式化
- ④ まとめ



## 評価文脈で困る例：必要呼び $\lambda$ 計算の決定性

```
Lemma answer_or_stuck_or_reducible M :  
  answer M  
  \/ (exists E x,  
      evalctx E /\ M = E.[tvar x] /\ bv E <= x)  
  \/ (exists E L N,  
      evalctx E /\ M = E.[L] /\ reduceI L N)  
  \/ (exists E L N,  
      evalctx E /\ M = E.[L] /\ reduceVCA L N).
```

■  $M$  についての帰納法で証明を試みる

## 必要呼びλ計算の決定性： $M = x$ のケース

```
x : var
=====
answer M
\ / (exists E y,
    evalctx E /\ tvar x = E.[tvar y]
    /\ bv E <= y)
\ / (exists E L N,
    evalctx E /\ tvar x = E.[L] /\ reduceI L N)
\ / (exists E L N,
    evalctx E /\ tvar x = E.[L]
    /\ reduceVCA L N)
```

■  $E$  が  $\square$  のとき，明らかに成り立つ

## 必要呼びλ計算の決定性： $M = x$ のケース

### ■ しかし，自動証明に失敗してしまう

```
Coq < eauto.  
x : var  
=====  
answer M  
\ / (exists E y,  
    evalctx E /\ tvar x = E.[tvar y]  
    /\ bv E <= y)  
\ / (exists E L N,  
    evalctx E /\ tvar x = E.[L] /\ reduceI L N)  
\ / (exists E L N,  
    evalctx E /\ tvar x = E.[L]  
    /\ reduceVCA L N)
```

## なぜ自動証明に失敗するか？

```
exists E y, evalctx E /\  
  tvar x = E.[tvar y] /\ bv E <= y
```

を証明するためには，

```
tvar x = E.[tvar y]
```

となるような  $E$  を探さなくてはならない



(一般的には) 高階パターンマッチングが必要

## 解決策：評価文脈の除去

### ■ 評価文脈のルールを簡約規則に展開

$$\bullet \xrightarrow{\beta}, \xrightarrow{\text{name}}, \xrightarrow{\text{I}} \text{と} \xrightarrow{\text{VCA}}$$

### ■ stuck 状態を表す述語の導入

$$\bullet \text{needs}_n(M, x) \ ( \iff \exists E.M = E_n[x] )$$
$$\text{と } \text{needs}(M, x) \ ( \iff \exists E.M = E[x] )$$

### ■ 評価規則

$$(V) \text{ let } x = V \text{ in } E[x] \rightarrow \text{let } x = V \text{ in } E[V]$$

を代入に簡略化

(元の定式化についても証明)

## 自動証明が回る例：必要呼び $\lambda$ 計算の決定性

```
Lemma answer_or_stuck_or_reducible M :  
  answer M \/  
    (exists x, needs M x) \/  
    (exists N, reduceI M N) \/  
    (exists N, reduceVCA M N).
```

Proof.

induction M as

[[? [Hanswer|[[[]|[[[]|[[[]]]]]

||? [Hanswer|[[[]|[[[]|[[[]]]]]

? [|[[[]]]|[[[]|[[[]]]]]]; eauto 6;

inversion Hanswer; subst; eauto 6.

Qed.

# アウトライン

- ① 対象言語：名前呼び  $\lambda$  計算と必要呼び  $\lambda$  計算
- ② 貢献 1：主定理の簡潔化された証明
- ③ 貢献 2：必要呼びラムダ計算の形式的検証に適した定式化
- ④ まとめ

## まとめ

- 必要呼び $\lambda$ 計算 [Ariola+ 1995] を検証に適するように変形し Coq で定式化
  - De Bruijn インデックス
  - 評価文脈の排除
- 名前呼びとの評価の対応に既存研究より簡単な証明を考案, Coq で定式化
  - 標準化定理
- 本研究の成果は FLOPS 2018 に投稿中
- TPP 2017 でも口頭発表