2026年3月10日
第28回プログラミングおよびプログラミング言語ワークショップ

shift0/reset0 を含む型付き言語における CPS変換の正当性の証明

田村優衣, 浅井健一
お茶の水女子大学

限定継続演算子の1つ

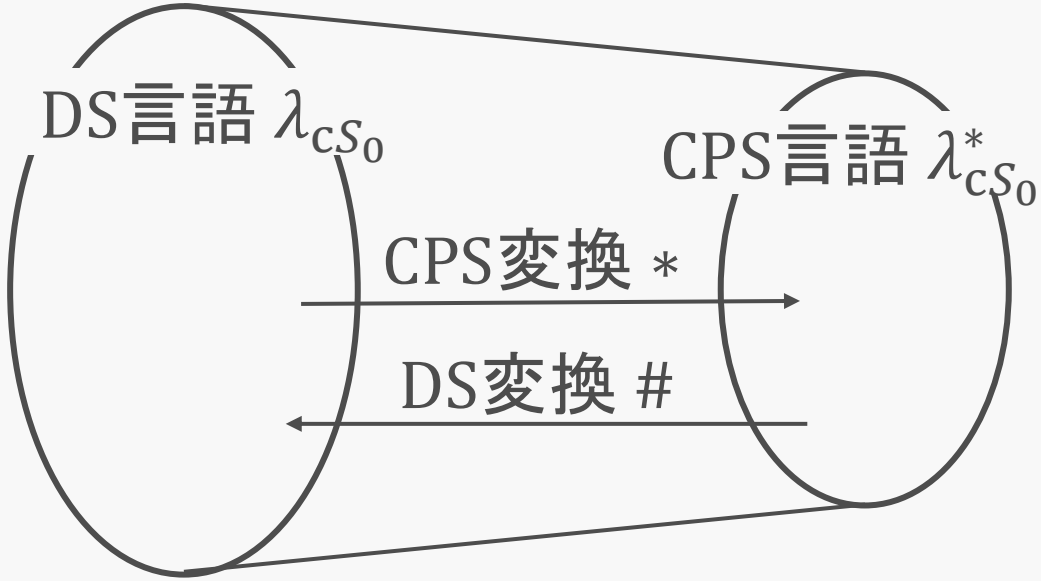
- **継続**
  - その後の計算
  - 例: $4 + [5 * 2] - 1$
    - 現在の計算：$5 * 2$
    - 現在の継続：$4 + [\cdot] - 1$
- **限定継続**
  - 範囲を限定した継続
  - reset $\langle \cdot \rangle$ で、切り取る範囲を表す
  - 例: $\langle 4 + [5 * 2] \rangle - 1$
    - 現在の計算：$5 * 2$
    - 現在の継続：$4 + [\cdot]$

代数的エフェクトハンドラの理論を考えるため

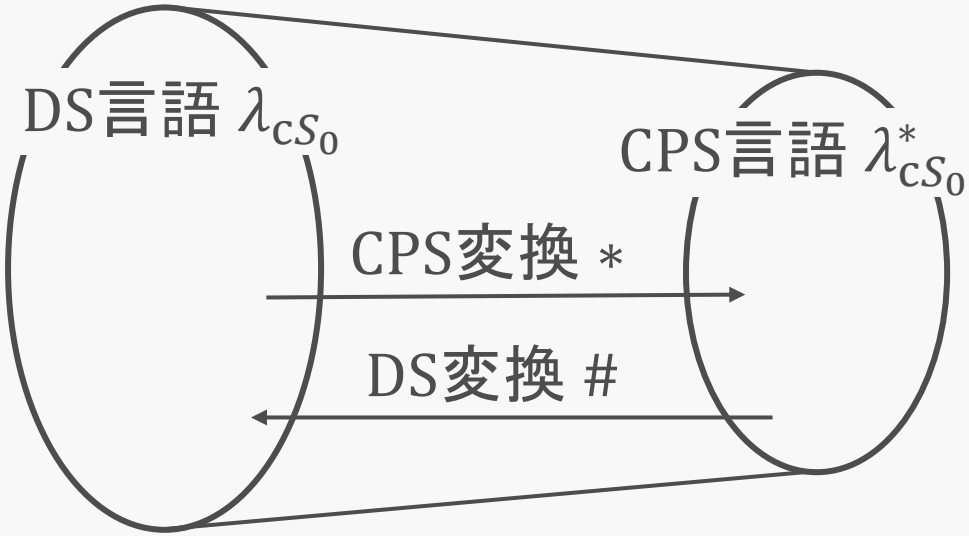
- **代数的エフェクトハンドラ（Algebraic Effects and Handlers, AEH）**
  - 限定継続を扱える言語機構の1つ
  - deep handler は shift0/reset0 と振る舞いが似ている [Forster et al. 2017]
    - 同じ理論的枠組みを用いた定式化 [Asai and Fujii, 2025]
- **AEHのdeep handlerを含む体系で reflection を証明したい**
  - そのために、shift0/reset0 を含む体系で reflection を証明する
  - 本成果は、reflection の証明の一環

2つの言語とその間の変換についての性質

- **どのような性質か？**
  - 2つの言語間の一対一対応を示す
  - 変換後に行える最適化が、変換前にも行える
- **本研究における対象**
  - DS言語
    - shift0/reset0 入りの単純型付きλ計算
  - CPS言語
    - リスト入りの単純型付きλ計算



4つの条件を満たすことを示す



- **DS変換とCPS変換の可逆性**

(1) DS の項 M に対して、$M \longrightarrow\!\!\!\!\!\rightarrow_{\lambda_{cS_0}} M^{*\#}$

(2) CPS の項 N に対して、$N \equiv_{\lambda^*_{cS_0}} N^{\#*}$

- **CPS変換の簡約の保存**

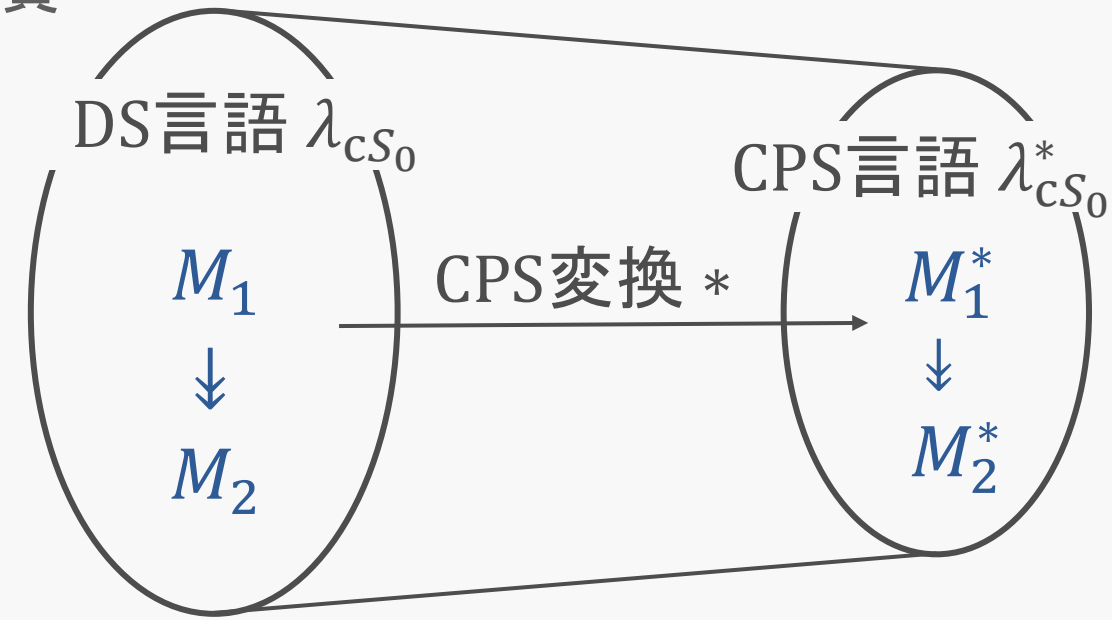
(3) DS の項 M_1, M_2 に対して、$M_1 \longrightarrow\!\!\!\!\!\rightarrow_{\lambda_{cS_0}} M_2$ ならば、${M_1}^* \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cS_0}} {M_2}^*$

- **DS変換の簡約の保存**

(4) CPS の項 N_1, N_2 に対して、$N_1 \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cS_0}} N_2$ ならば、${N_1}^{\#} \longrightarrow\!\!\!\!\!\rightarrow_{\lambda_{cS_0}} {N_2}^{\#}$

reflection（3）：CPS変換の簡約の保存を証明

- **CPS変換の簡約の保存**
  - DS言語における簡約とCPS言語における簡約が対応する
  - 正当性（=等価性の保存）よりも強い性質
- **この証明に必要な準備**
  - 各言語の定式化
    - DS言語
    - CPS言語
  - CPS変換の定義



比較が容易な shift/reset の手法を参考にする

- **shift0/dollar [Biernacki et al, 2022]**
  - 継続をスタックとして扱う手法（本研究のリスト構造と類似）
  - 独自の枠組みを入れている

    → 対象（shift0/reset0）との比較が困難

- **shift/reset [Biernacki et al, 2020] [本田ら, 2023]**
  - shift0/reset0 と同じ枠組みで扱える [Ishio et al, 2022]

    → 対象（shift0/reset0）の体系へ素直に拡張できる

shift0 はメタ継続を捕捉できる

- **shift** $Sk.e$
  - 簡約規則 $\langle J[Sk.e]\rangle \to \langle e[\lambda x.\langle J[x]\rangle/k]\rangle$
- **shift0** $S_0k.e$
  - 簡約規則 $\langle J[S_0k.e]\rangle \to e[\lambda x.\langle J[x]\rangle/k]$
  - 「より外側の継続」を捕捉できる
    - これを**メタ継続**と呼ぶ

$$\langle\ (\lambda x.1) \mathbin{@} \langle\ (\lambda y.2) \mathbin{@} S_0k_1.S_0k_2.(k_1 \mathbin{@} 0)\ \rangle\ \rangle$$

$$\begin{aligned}
& \langle (\lambda x.1) \mathbin{@} \langle (\lambda y.2) \mathbin{@} Sk_1.Sk_2.(k_1 \mathbin{@} 0)\rangle\rangle \\
\longrightarrow\ & \langle (\lambda x.1) \mathbin{@} \langle\, Sk_2.(k_1 \mathbin{@} 0)[\lambda z.\langle (\lambda y.2) \mathbin{@} z\rangle/k_1]\,\rangle\rangle \\
\longrightarrow\ & \langle (\lambda x.1) \mathbin{@} \langle Sk_2.((\lambda z.\langle (\lambda y.2) \mathbin{@} z\rangle) \mathbin{@} 0)\rangle\rangle \\
\longrightarrow\ & \langle (\lambda x.1) \mathbin{@} \langle ((\lambda z.\langle (\lambda y.2) \mathbin{@} z\rangle) \mathbin{@} 0)[\lambda z.\langle z\rangle/k_2]\rangle\rangle \\
\longrightarrow\ & \ldots\text{(中略)} \\
\longrightarrow\ & 1
\end{aligned}$$

$$\begin{aligned}
& \langle (\lambda x.1) \mathbin{@} \langle (\lambda y.2) \mathbin{@} S_0k_1.S_0k_2.(k_1 \mathbin{@} 0)\rangle\rangle \\
\longrightarrow\ & \langle (\lambda x.1) \mathbin{@} (\,S_0k_2.(k_1 \mathbin{@} 0)[\lambda z.\langle (\lambda y.2) \mathbin{@} z\rangle/k_1]\,)\rangle \\
\longrightarrow\ & \langle (\lambda x.1) \mathbin{@} (S_0k_2.((\lambda z.\langle (\lambda y.2) \mathbin{@} z\rangle) \mathbin{@} 0))\rangle \\
\longrightarrow\ & ((\lambda z.\langle (\lambda y.2) \mathbin{@} z\rangle) \mathbin{@} 0)[\lambda z.\langle (\lambda x.1) \mathbin{@} z\rangle/k_2] \\
\longrightarrow\ & \ldots\text{(中略)} \\
\longrightarrow\ & 2
\end{aligned}$$

shift/reset に対する手法をメタ継続で拡張する

1. **各言語の定式化**
   - DS言語
   - CPS言語
2. **各変換の定義**
   - CPS変換（コロン変換）（一部のみ紹介）
   - DS変換
3. **reflection の証明**
   - 4つの条件を示す
     - (3) CPS変換の簡約の保存
     - (1) (2) (4)

shift/reset を含む定式化に shift0 を追加

単純型付き
λ計算

＋

- let文
- shift
- shift0
- reset

定数として扱う

$$
\begin{array}{rlll}
\text{terms} & L, M, N & ::= & V \mid P \\
\text{values} & V, W & ::= & x \mid \underline{\lambda} x.\, M \mid \underline{\mathcal{S}} \mid \underline{\mathcal{S}_0} \\
\text{nonvalues} & P, Q & ::= & M \mathbin{\underline{@}} N \mid \underline{\mathsf{let}}\ x = M\ \underline{\mathsf{in}}\ N \mid \underline{\langle M \rangle} \\
\text{pure contexts} & J, K & ::= & [\,] \mid K[[\,] \mathbin{\underline{@}} M] \mid K[V \mathbin{\underline{@}} [\,]] \mid K[\underline{\mathsf{let}}\ x = [\,]\ \underline{\mathsf{in}}\ M]
\end{array}
$$

継続とメタ継続を扱う2CPSの形

root	R	::=	$\underline{\lambda} k.\, \underline{\lambda} g.\, M_k$
terms	$M_{(\Delta{+}{+}\Theta)}$	::=	$K_\Delta \mathbin{\underline{@}} V \mathbin{\underline{@}} G_\Theta \mid V \mathbin{\underline{@}} W \mathbin{\underline{@}} K_\Delta \mathbin{\underline{@}} G_\Theta$
values	V, W	::=	$x \mid (\underline{\lambda} x.\, \underline{\lambda} k.\, \underline{\lambda} g.\, M_k) \mid S \mid S_0$
shift	S	::=	$\underline{\lambda} w.\, \underline{\lambda} j.\, \underline{\lambda} g.\, w \mathbin{\underline{@}} (\underline{\lambda} y.\, \underline{\lambda} k.\, \underline{\lambda} g.\, j \mathbin{\underline{@}} y \mathbin{\underline{@}} (k :\!\!: g)) \mathbin{\underline{@}} k_{id} \mathbin{\underline{@}} g$
shift0	S_0	::=	$\underline{\lambda} w.\, \underline{\lambda} j.\, \underline{\lambda} (k_0 :\!\!: g).\, w \mathbin{\underline{@}} (\underline{\lambda} y.\, \underline{\lambda} k.\, \underline{\lambda} g.\, j \mathbin{\underline{@}} y \mathbin{\underline{@}} (k :\!\!: g)) \mathbin{\underline{@}} k_0 \mathbin{\underline{@}} g$
continuations	J_Δ, K_Δ	::=	${}_{(\Delta=k)}\, k \mid_{(\Delta=\bullet)} k_{id} \mid \underline{\lambda} x.\, \underline{\lambda} g.\, M_\Delta$
meta continuations	G_Θ	::=	${}_{(\Theta=g)}\, g \mid_{(\Theta=\Delta)} G_\Delta$
non-empty metaconts	$G_{(\Delta{+}{+}\Theta)}$	::=	$K_\Delta \mathbin{\underline{:\!\!:}} G_\Theta$

メタ継続は継続のリスト

継続 k の位置を追跡する

- 継続 k がメタ継続に押し出される

$$\underline{\langle} 1 \underline{\rangle}^* = \underline{\lambda} k. \underline{\lambda} g. (\; k_{id} \; \underline{@} \, 1 \, \underline{@} \; (k \mathbin{\underline{::}} g) \;)$$

- 継続とメタ継続の状態を Δ と Θ で表す

Δ++Θ の定義

$$\begin{array}{cccc} \Delta & ++ & g & = & \Delta \\ \bullet & ++ & \Delta & = & \Delta \end{array}$$

継続 K_Δ の Δ

- Δ = k : 継続部分に k がある
- Δ = ・ : 継続部分に k がない

メタ継続 G_Θ の Θ

- Θ = g : メタ継続なし
- Θ = Δ : メタ継続あり（Δ = k or ・）

5つの補題を用いて Agda で証明

- **定理**

  DS の項 M_1, M_2 に対して、$M_1 \longrightarrow\!\!\!\!\!\rightarrow_{\lambda_{cs_0}} M_2$ ならば、${M_1}^* \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cs_0}} {M_2}^*$

- **3つの難所**
  - 補題2（継続の代入補題）
    - 代入先である継続 k の位置をどう特定するか？
  - 補題4（継続とメタ継続の簡約の保存）
    - 止まってしまう帰納法をどう回すか？
  - 補題5（コンテキストの存在補題）
    - メタ継続がどのように関係するか？

継続 k の現れる位置を Δ と Θ の場合分けで特定

- k がどこにあるか
  - 継続部分： Δ = k, Θ = g
  - メタ継続部分 : Δ = ▪, Θ = k

$$(\underline{\lambda} x.\, x)^* = \underline{\lambda} k.\, \underline{\lambda} g.\, (\, k \;\underline{@}\, x \,\underline{@}\; g \,)$$

$$\underline{\langle 1 \rangle}^* = \underline{\lambda} k.\, \underline{\lambda} g.\, (\, k_{id} \;\underline{@}\, 1 \,\underline{@}\; (k :: g) \,)$$

補題 2 (継続の代入補題, `Reflect3.agda` の `lemma-csubst`). 任意の $\lambda_{c\mathcal{S}_0}$ の項 M と値 V, $\lambda^*_{c\mathcal{S}_0}$ の継続 K_Δ, J_Δ, メタ継続 G_Θ について、以下が成り立つ。

- $\Delta = k, \Theta = g$ ならば $M : (K_k[J_\Delta/k]) : G_g = (M : K_k : G_g)[J_\Delta/k]$
- $\Delta = \bullet, \Theta = k$ ならば $M : K_\bullet : (G_k[J_\Delta/k]) = (M : K_\bullet : G_k)[J_\Delta/k]$

原因はコロン変換の挙動。2つ命題の統合により解決

- 「継続の命題」と「メタ継続の命題」を1つの補題に統合

補題 4 (`Reflect3.agda` の `correctCM`). 任意の λ_{cS_0} の項 M について、以下が成り立つ。

- 任意の $\lambda^*_{cS_0}$ の継続 K_Δ, K'_Δ とメタ継続 G_Θ について $K_\Delta \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cS_0}} K'_\Delta$ ならば $M : K_\Delta : G_\Theta \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cS_0}} M : K'_\Delta : G_\Theta$
- 任意の $\lambda^*_{cS_0}$ の継続 K_Δ とメタ継続 G_Θ, G'_Θ について $G_\Theta \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cS_0}} G'_\Theta$ ならば $M : K_\Delta : G_\Theta \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{cS_0}} M : K_\Delta : G'_\Theta$

- 本研究で導出したコロン変換
  - reset 付きの項に対する変換で、継続がメタ継続の先頭に追加される

$$\underline{\langle} M \underline{\rangle} : K : G = M : k_{id} : (K \mathbin{\underline{::}} G)$$

  - 2つの命題を別々にすると、帰納法が回らなくなる

メタ継続は関与しない

- **shift, shift0 に関する重要補題**

補題 5 (`Reflect3.agda` の `contExist`). 任意の $\lambda_{c\mathcal{S}_0}$ のコンテキスト J と $\lambda^*_{c\mathcal{S}_0}$ の継続 K_Δ について、以下の 2 つを満たすような $\lambda^*_{c\mathcal{S}_0}$ の継続 J'_Δ が存在する。

- 任意の $\lambda_{c\mathcal{S}_0}$ の値でない項 P と $\lambda^*_{c\mathcal{S}_0}$ のメタ継続 G_Θ について、$J[P] : K_\Delta : G_\Theta = P : J'_\Delta : G_\Theta$
- 任意の $\lambda_{c\mathcal{S}_0}$ の値 V と $\lambda^*_{c\mathcal{S}_0}$ のメタ継続 G_Θ について、$V : J'_\Delta : G_\Theta \longrightarrow\!\!\!\!\!\rightarrow_{\lambda^*_{c\mathcal{S}_0}} J[V] : K_\Delta : G_\Theta$

- **メタ継続による拡張**
  - 数式中の G_Θ は、両辺で変化せずにそのまま引き回されている
  - shift も shift0 も同じ補題を使って証明できた

まとめ

- **本研究の成果**
  - CPS言語の定式化をメタ継続が入った形に拡張
    - 継続とメタ継続に Δ と Θ を導入し、挙動を表現
  - コロン変換の導出
    - Danvyらの手法を、shift0/reset0 入りのインタプリタに適応
  - 「CPS変換の簡約の保存」を Agda で証明
- **今後の展望**
  - 残りの3つ条件を証明し、reflection が成り立つことを示す
  - AEH の deep handler 入りの体系で reflection を証明する