Simulating β-reduction in Combinatory Logic

Naosuke Matsuda (Tokyo-Tech)

joint work with Yuichi Komori, Fumika Yamakawa

Contents

- Motivation (λ-calculus and α-conversion)
- Consideration
- Canonical Expression of $\Lambda/{=_\alpha}$-Terms
- Simulating β-reductions through CL
- Future Work

Motivation: λ-calculus and α-conversion

λ-calculus and α-conversion

Def (λ-term)

$$x \in V : \quad x ::= x_1 \mid x_2 \mid \cdots$$
$$F \in \Lambda : \quad F ::= x \mid (FF) \mid (\lambda x. F)$$

Parenthesis are omitted as follows.

$$F_1 F_2 F_3 \equiv \left((F_1 F_2) F_3\right)$$
$$\lambda xy. F \equiv \left(\lambda x. (\lambda y. F)\right)$$

λ-calculus and α-conversion

Def (β-reduction)

$$(\lambda x. F)G \to_{1\beta} [G/x]F$$

(where $x \in FV(G)$ is not bound in F)

$$\frac{F_1 \to F_2}{GF_1 \to_{1\beta} GF_2} \quad \frac{F_1 \to F_2}{F_1 G \to_{1\beta} F_2 G} \quad \frac{F_1 \to_{1\beta} F_2}{\lambda x. F_1 \to_{1\beta} \lambda x. F_2}$$

$\to_\beta$: reflexive transitive closure of $\to_{1\beta}$

$=_\beta$: smallest equivalent relation which contains $\to_{1\beta}$

λ-calculus and α-conversion

$$(\lambda xy.xy)y \to_{1\beta} \lambda y.yy$$

$$\downarrow_\alpha$$

$$(\lambda xz.xz)y \to_{1\beta} \lambda z.yz$$

Def (α-conversion)

$\to_\alpha$: converting some bound variables

$=_\alpha$: smallest equivalence relation which contains $\to_\alpha$

λ-calculus and α-conversion

$\lambda\beta$ is the pair $\langle \Lambda, =_\alpha, =_\beta \rangle$.

There are many problems relating to $=_\alpha$.

How can we:

- implement the α-conversion operation?
- decide the relation $=_\alpha$?

λ-calculus and α-conversion

Solution strategy for this problem

- canonical representation of terms

  (de Bluijn index, abstraction with maps...)

- $V = FV \uplus BV$

  (providing two sorts of variables)

- reconstructing the theory

  with combinatory terms

⋮

λ-calculus and α-conversion

Def (Combinatory Term, weak reduction)

$$C \in \mathrm{CT}: \quad C ::= x \mid \mathrm{S} \mid \mathrm{K} \mid (CC)$$

$$\mathrm{K}CD \rightarrow_{1w} C$$

$$SCDE \rightarrow_{1w} CE(DE)$$

$$\frac{F_1 \rightarrow_{1w} F_2}{GF_1 \rightarrow_{1w} GF_2} \quad \frac{F_1 \rightarrow_{1w} F_2}{F_1G \rightarrow_{1w} F_2G} \quad \frac{F_1 \rightarrow_{1w} F_2}{\lambda x.F_1 \rightarrow_{1w} \lambda x.F_2}$$

λ-calculus and α-conversion

Def $(\lambda^* x. C)$

$$\lambda^* x. x \equiv \mathrm{SKK}$$
$$\lambda^* x. C \equiv \mathrm{K}C \qquad \text{if } x \notin FV(C)$$
$$\lambda^* x. CD \equiv \mathrm{S}(\lambda^* x. C)(\lambda^* x. D)$$

Note that x does not occur in $\lambda x^*.C$.

Theorem

$$(\lambda^* x. C)D \rightarrow_w [D/x]C$$

λ-calculus and α-conversion

We can obtain the term $\lambda x^*.x \equiv \mathrm{SKK}$ which do the same work as $\lambda x.x$ without using x.

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

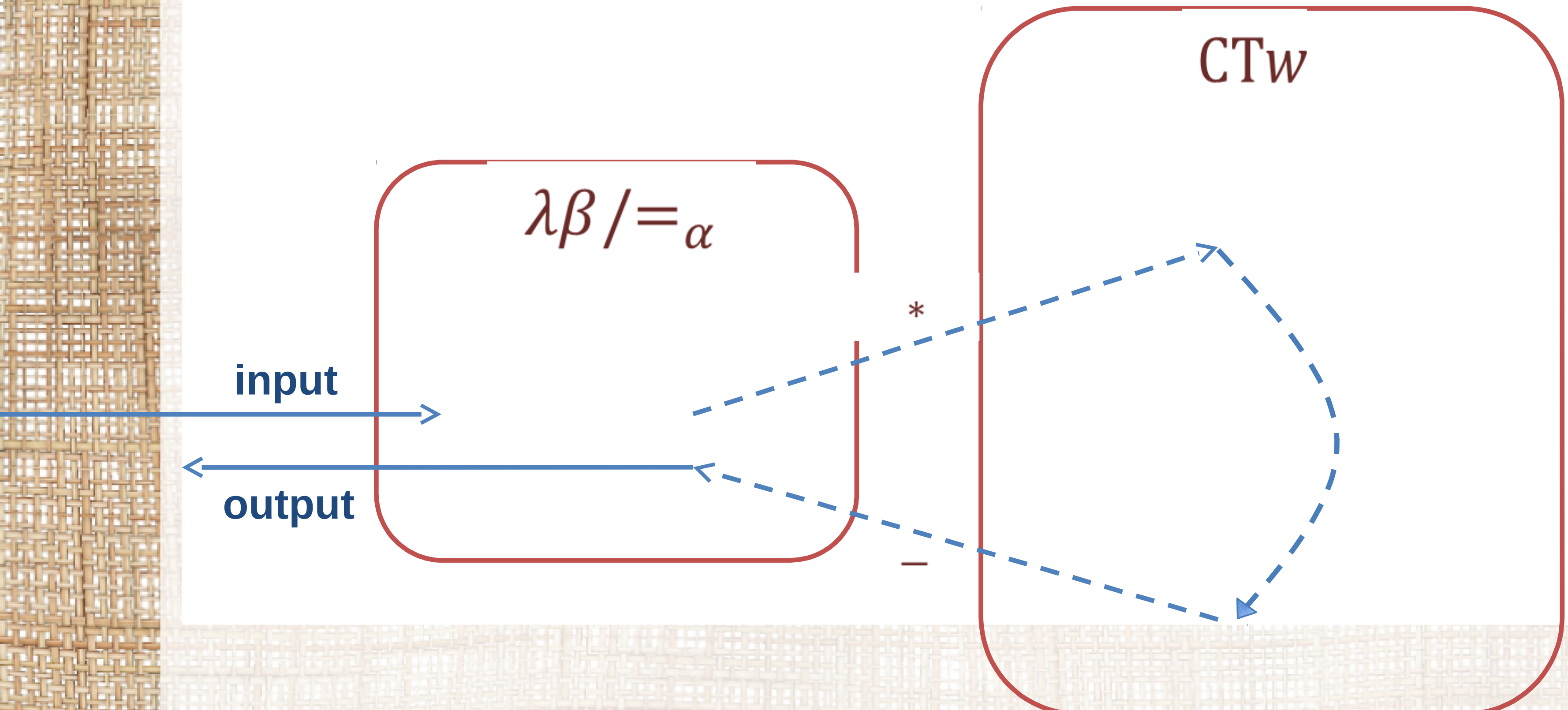
S K ((

For this technical advantage, we have to sacrifice the intuitive clarity of the λ-notation.

$$\lambda^* x.xyy \equiv \mathrm{S(S(SKK)(K}y\mathrm{))(K}y\mathrm{)}$$

λ-calculus and α-conversion

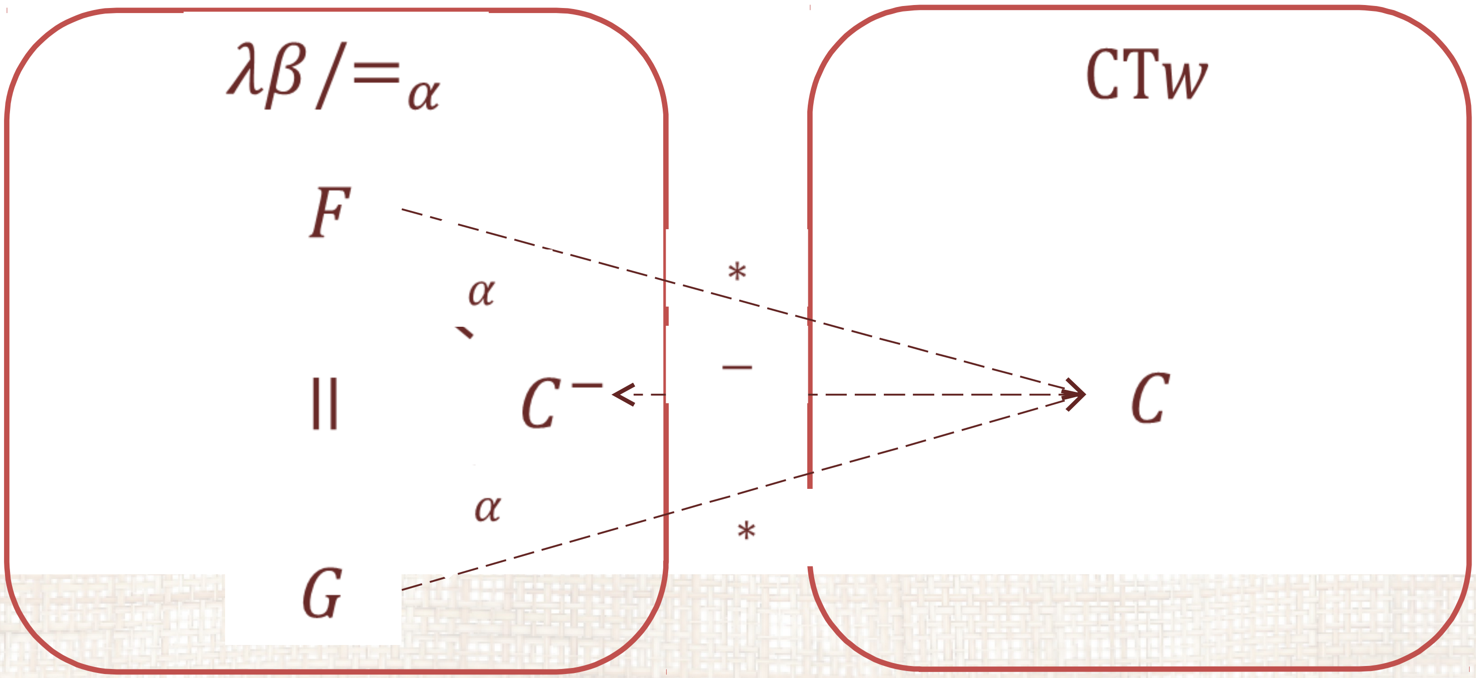
Therefore, we try to use CTw as a simulater which simulates $\lambda\beta/{=_\alpha}$, and use λ-terms to display the values.



λ-calculus and α-conversion

Aim 1

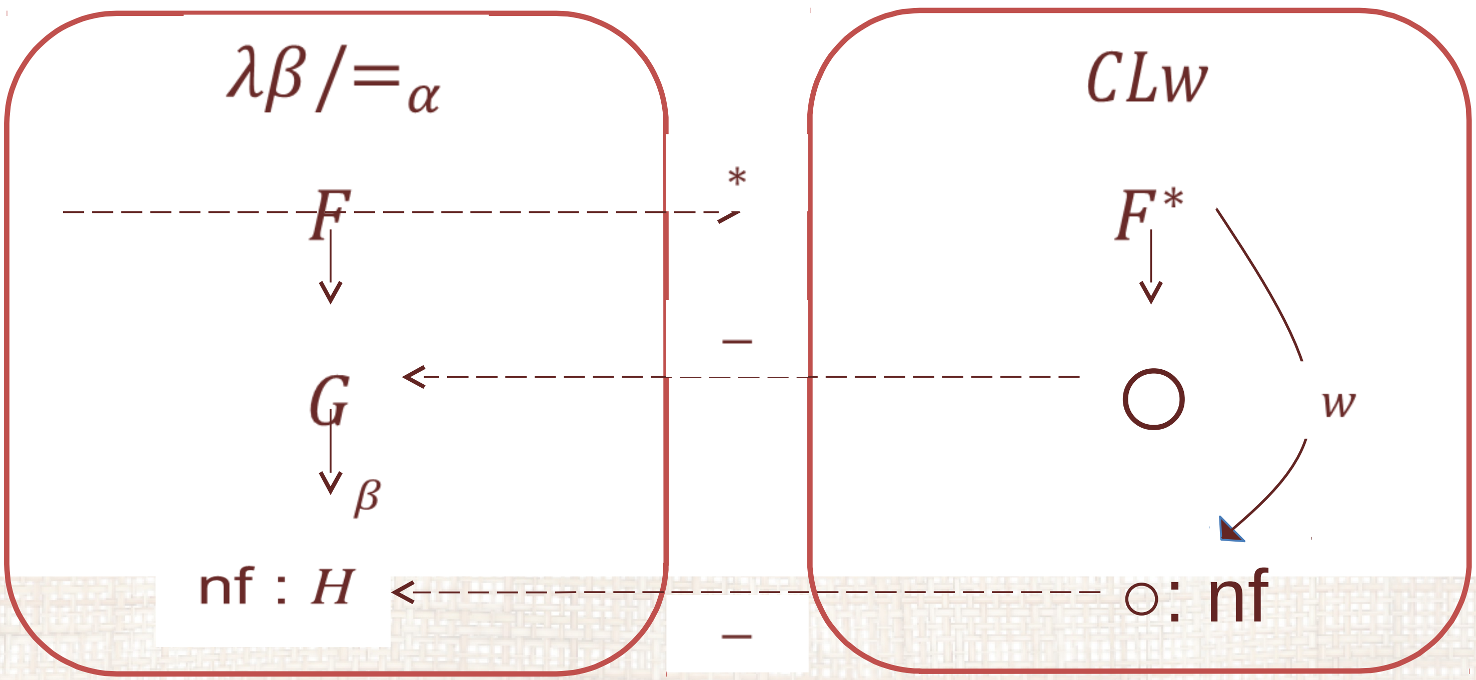
$$(A1)\ \ (F^*)^- =_\alpha F \quad (\text{and } F =_\alpha G \Rightarrow (F^*)^- \equiv (G^*)^-)$$



λ-calculus and α-conversion

Aim 2

$$(A2)\ F \to_\beta G \Rightarrow \exists C \text{ s.t. } F^* \to_w C,\ C^- =_\alpha G$$



Consideration

Consideration

Many methods are proposed for such simulation. But most of them are introduced to simulate the $\Lambda/{=_{\alpha\beta}}$-theory in CL:

$$(F^*)^- =_{\alpha\beta} F$$

$$F =_\beta G\,? \quad \longleftrightarrow \quad F^* =_w G^*?$$

But, our aim is to simulate more precisely.

Consideration

Example (natural interpretation)

$$x^C \equiv x \quad (FG)^C \equiv F^C G^C \quad (\lambda x.F)^C \equiv \lambda^* x.F^C$$

$$x^\lambda \equiv x \quad \mathrm{K}^\lambda \equiv \lambda xy.x$$
$$\mathrm{S}^\lambda \equiv \lambda xyz.xz(yz) \quad (CD)^\lambda \equiv C^\lambda D^\lambda$$

Theorem

(1) $(F^C)^\lambda \to_{\alpha\beta} F$

(2) $F \to_\beta G \Rightarrow F^C \to_w G^C$

Consideration

Example

$$(\lambda x.y)^C \equiv \mathrm{K}y,$$

but

$$(Ky)^\lambda =_\alpha (\lambda zx.z)y \ \left(\to_{1\beta} \lambda x.y\right).$$

It is provable that this gap is caused by a difference in arities: Lambda abstraction $\lambda x.F$ works immediately when it gets one object, but combinator K (or S) only works when it gets two (or three) objects.

Canonical Expression of $\Lambda/{=_\alpha}$-terms

Canonical Expression of $\Lambda/{=_\alpha}$-terms

To achieve the aim

$$(A1)\ (F^*)^- =_\alpha F,$$

we introduce a new combinator I_λ. That is, the definition of combinatory terms is extended as follows:

$$C \in \mathrm{CT} \quad C ::= x \mid \mathrm{S} \mid \mathrm{K} \mid \mathrm{I}_\lambda \mid (CC)$$

This idea was introduced by Komori-Yamakawa, and they showed that this combinator enable us to achieve $(A1)$.

Canonical Expression of $\Lambda/{=_\alpha}$-terms

Def

$$x^* \equiv x \quad (\lambda x.F)^* \equiv I_\lambda(\lambda^* x.F^*) \quad (FG)^* \equiv F^* G^*$$

$$x^- \equiv x \quad \mathrm{K}^- \equiv \lambda xy.x$$

$$\mathrm{S}^- \equiv \lambda xyz.xz(yz) \quad (CD)^- \equiv C^- D^-$$

$$(\mathrm{I}_\lambda C)^- \equiv \lambda x.D^- \;\; (x \notin FV(C),\ D \text{ is } w\text{-nf of } Cx)$$

Note that $-$ is partial (from Λ^* into Λ).

Canonical Expression of $\Lambda/{=_\alpha}$-terms

Theorem (Komori-Yamakawa 2011)

$$(F^*)^- =_\alpha F$$

$$(\lambda x.F)^* \equiv \mathrm{I}_\lambda(\lambda^* x.F^*)$$

$$(\mathrm{I}_\lambda(\lambda^* x.F^*))^- =_\alpha \lambda x.(F^*)^- =_\alpha \lambda x.F$$

$$(\lambda^* x.F^*)x \to_w F^*: \text{w-nf}$$

Simulating β-reductions through CL

Simulating β-reductions through CL

Considering the reduction rule, there are some problems caused by I_λ.

(1) $(\lambda x.F)G \rightarrow_{1\beta} [G/x]F$, but:

$$\mathrm{I}_\lambda(\lambda x.F^*)G^* \rightarrow_w [G^*/x]F^*$$

I_λ blocks the intended reductions

(2) $\lambda x.\big((\lambda y.yx)x\big) \rightarrow_{1\beta} \lambda x.xx$, but:

$$(\lambda x.(\lambda y.yx)x)^* \equiv I_\lambda\big(\lambda^* x.\big((\lambda^*.yx)x\big)^*\big)$$

$$\equiv \mathrm{I}_\lambda\Big(\mathrm{S}\Big(\mathrm{S}(\mathrm{K}\mathrm{I}_\lambda)\Big(\mathrm{S}\big(\mathrm{K}(\mathrm{SKK})\big)\Big)\mathrm{K}\Big)\Big)(\mathrm{SKK})$$

λ^* disarranges the form of its inner term

and blocks the intended reductions

disarranges the form of its inner term

Simulating β-reductions through CL

To get over this problem, we introduce a new combinator L and give the following reduction relation on CT.

$$\mathrm{K}CD \to_1 C$$

$$SCDE \to_1 CE(DE)$$

$$\mathrm{I}_\lambda CD \to_1 CD \quad \cdots \text{ (A)}$$

$$\mathrm{I}_\lambda C \to_1 \mathrm{L}x(Cx) \quad (x \notin FV(C)) \quad \cdots \text{ (B)}$$

$$\frac{F_1 \to_1 F_2}{GF_1 \to_1 GF_2} \quad \frac{F_1 \to_1 F_2}{F_1G \to_1 F_2G} \quad \frac{F_1 \to_1 F_2}{\lambda x.F_1 \to_1 \lambda x.F_2}$$

Simulating β-reductions through CL

I_λ-reduction (A) removes I_λ and enable us to continue our calculation.

$$(\lambda x.F)G \rightarrow_{1\beta} [G/x]F$$

$$\begin{aligned} I_\lambda(\lambda^* x.F^*)G^* &\rightarrow_1 (\lambda^* x.F^*)G^* \\ &\rightarrow [G^*/x]F^* \end{aligned}$$

Simulating β-reductions through CL

I_λ-reduction (B) arranges the term of the form $\lambda^* x. F^*$, and enable us to continue our calculation.

$$\lambda x. F \to_{1\beta} \lambda x. G$$

$$\begin{aligned} \mathrm{I}_\lambda(\lambda^* x. F^*) &\to_1 Lx\left((\lambda^* x. F^*)x\right) \\ &\to \ Lx\,(F^*) \\ &\to \ Lx\,(G^*) \end{aligned}$$

Simulating β-reductions through CL

Because of the work of L-combinator, we have to extend the definition of $-$ as follows:

$$x^- \equiv x \quad \mathrm{K}^- \equiv \lambda xy.x$$

$$\mathrm{S}^- \equiv \lambda xyz.xz(yz) \quad (CD)^- \equiv C^- D^-$$

$$(\mathrm{I}_\lambda C)^- \equiv \lambda x.D^- \ \ (x \notin FV(C),\ D \text{ is } w\text{-nf of } Cx)$$

$$(LxC)^- \equiv \lambda x.C^-$$

Simulating β-reductions through CL

Example

$$\lambda x.\left((\lambda y.y)z\right) \to_\beta \lambda x.z$$

$$\begin{aligned}
I_\lambda\left(\lambda^* x.\left((\lambda y.y)z\right)^*\right) &\to_1 Lx\left(\left(\lambda^* x.\left((\lambda y.y)z\right)^*\right)x\right) \\
&\to \; Lx\left((\lambda y.y)z\right)^* \\
&\equiv \; Lx\left((\lambda^* y.y)z\right) \\
&\to_1 Lx\left((\lambda^* y.y)z\right) \\
&\to \; Lxz
\end{aligned}$$

$$(Lxz)^- \equiv \lambda x.z$$

Simulating β-reductions through CL

Note that after we apply the rule $\mathrm{I}_\lambda C \to_1 \mathrm{L}x(Cx)$, we cannot transform the subterm $\mathrm{L}x$. But, with the standardization theorem, we can obtain the following result (aim $(A2)$).

Theorem

$$F \to G \Rightarrow \exists C \text{ s.t. } F^* \to C \text{ and } C^- =_\alpha G$$

Simulating β-reductions through CL

Especially, if $F \to_\beta G : \beta$-nf then we can get a term C s.t. $C^- =_\alpha G$ by following algorithm.

For F^*, do the following procedure until I_λ does not occur in term.

Take a leftmost I_λ-combinator. If the form is $\mathrm{I}_\lambda CD$ then we transform it into w-nf of CD. Else the form is $\mathrm{I}_\lambda C$, and we transform it into LxD where D is w-nf of Cx.

Future Work

Future Work

1.

Can we the same simulation without using L-combinator?

2.

How can we simulate an arbitrary reduction sequence?

Thank you for listening.